INFORMATICS PRACTICES

11149

TEXTBOOK FOR CLASS XI

© NCERT not to be republished

विद्यया ऽमृतमश्नुते
एन सी ई आर टी
NCERT

राष्ट्रीय शैक्षिक अनुसंधान और प्रशिक्षण परिषद्
NATIONAL COUNCIL OF EDUCATIONAL RESEARCH AND TRAINING

Reprint 2026-27

11149 – INFORMATICS PRACTICES
Textbook for Class XI

ISBN 978-93-5292-148-5

First Edition
August 2019 Shravana 1941

Reprinted
June 2021 Ashadha 1943
October 2022 Kartika 1944
March 2024 Chaitra 1946
February 2025 Magha 1946
February 2026 Phalguna 1947

PD 50T BS

© National Council of Educational Research and Training, 2019

₹ 170.00

Printed on 80 GSM paper with NCERT watermark

Published at the Publication Division by the Secretary, National Council of Educational Research and Training, Sri Aurobindo Marg, New Delhi 110016 and printed at Saraswati Art Printers, E-25, Sector-4, Bawana Industrial Area, Delhi -110 039

ALL RIGHTS RESERVED

- No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior permission of the publisher.
- This book is sold subject to the condition that it shall not, by way of trade, be lent, re-sold, hired out or otherwise disposed of without the publisher's consent, in any form of binding or cover other than that in which it is published.
- The correct price of this publication is the price printed on this page, Any revised price indicated by a rubber stamp or by a sticker or by any other means is incorrect and should be unacceptable.

OFFICES OF THE PUBLICATION DIVISION, NCERT

NCERT Campus
Sri Aurobindo Marg
New Delhi 110 016 **Phone : 011-26562708**

108, 100 Feet Road
Hosdakere Halli Extension
Banashankari III Stage
Bengaluru 560 085 **Phone : 080-26725740**

Navjivan Trust Building
P.O.Navjivan
Ahmedabad 380 014 **Phone : 079-27541446**

CWC Campus
Opp. Dhankal Bus Stop
Panihati
Kolkata 700 114 **Phone : 033-25530454**

CWC Complex
Maligaon
Guwahati 781 021 **Phone : 0361-2674869**

Publication Team

Head, Publication Division : *M.V. Srinivasan*

Chief Editor : *Bijnan Sutar*

Chief Business Manager : *Amitabh Kumar*

Chief Production Officer (In charge) : *Deepak Jaiswal*

Assistant Production Officer : *Sayuraj A.R.*

Cover Design and Layout
Meetu Sharma (Contractual)

© NCERT not to be republished

Foreword

Information Technology has continuously been crossing the barriers of access and communication and reaching more and more people. The number of internet users in India has been on the rise. The tremendous growth in computer science, telecommunications and information technology has resulted in automation of various tasks and contributed to the ease of living. Technology has made continuous inroads into diverse areas—be it business, commerce, science, sports, health, transportation or education. Today, we are living in an interconnected world where computer based applications influence the way we learn, communicate, commute, or even socialise.

With so many users of information and communication technology (ICT), huge volumes of data are continuously generated at an unprecedented rate. Many innovative business models are being evolved which utilise such data to reach potential customers in a more targeted way. Government agencies are also using data to deliver services and fast track progress of different programmes, strengthen accountability and to make more informed decisions. This has been creating better opportunities for our youth not only to enter the field of technical education but also in the world of work. NCERT, for the first time, has developed a textbook on 'Informative Practices' to develop skill sets in students to make use of the opportunities provided by ICT.

This book focuses on the fundamental concepts related to handling of data while opening a window to the emerging areas of data processing. It seeks to address the dual challenges of reducing curricular load as well as introducing the latest development in the field of ICT.

As an organisation committed to systemic reforms and continuous improvement in the quality of its curricular material, NCERT welcomes comments and suggestions to enable us to bring about necessary changes in its further publications.

Hrushikesh Senapaty
Director
National Council of Educational
Research and Training

New Delhi
July 2019

© NCERT not to be republished

THE CONSTITUTION OF INDIA

PREAMBLE

WE, THE PEOPLE OF INDIA, having solemnly resolved to constitute India into a [1]**[SOVEREIGN SOCIALIST SECULAR DEMOCRATIC REPUBLIC]** and to secure to all its citizens :

JUSTICE, social, economic and political;

LIBERTY of thought, expression, belief, faith and worship;

EQUALITY of status and of opportunity; and to promote among them all

FRATERNITY assuring the dignity of the individual and the [2][unity and integrity of the Nation];

IN OUR CONSTITUENT ASSEMBLY this twenty-sixth day of November, 1949 do **HEREBY ADOPT, ENACT AND GIVE TO OURSELVES THIS CONSTITUTION.**

---

1. Subs. by the Constitution (Forty-second Amendment) Act, 1976, Sec.2, for "Sovereign Democratic Republic" (w.e.f. 3.1.1977)
2. Subs. by the Constitution (Forty-second Amendment) Act, 1976, Sec.2, for "Unity of the Nation" (w.e.f. 3.1.1977)

Preface

In the present education system of our country, specialised/discipline based courses are introduced at the higher secondary stage. This stage is crucial as well as challenging because of the transition from general to discipline-based curriculum. The syllabus at this stage needs to have sufficient rigour and depth while remaining mindful of the comprehension level of the learners. Further, the textbook should not be heavily loaded with content.

We are living in an era where information drives many of our socio-economic decisions. Millions of people are accessing internet round the clock for availing various services and thereby generating vast amount of data. Processing of data is becoming a key skill with applications across the disciplines. Thus, study of basic concepts of data handling and analysis is becoming more and more desirable. There are courses offered in the name of computer science, Information and Communication Technology (ICT), Information Technology (IT), etc. by various boards and schools up to the secondary stage, as an optional subject. These mainly focus on using computer for word processing, presentation tools and application software.

Informatics Practices (IP) at the higher secondary stage of school education is also offered as an optional subject. At this stage, students can take up IP with the aim of pursuing a career in data science or related areas after going through professional courses at higher levels. Therefore, at the higher secondary stage, the curriculum of IP introduces the basics of database management systems and data processing. The book has eight chapters covering the following broader themes:

- Basic understanding of computer systems and their evolution, introduction to software and their categorisation, computer memory, awareness of emerging trends in the field of information and communication technology.
- Basic constructs of a program using Python programming language — program structure, identifiers, variables, flow of control, advanced data types like Lists and Dictionaries.
- Handling data using specialised Python library called NumPy — concept of single and multi-dimensional Array.
- Concepts of data, database, and relational database management system using MySQL. Structured query language — data definition, data manipulation and data querying.

Python programming language and NumPy are introduced using both the interactive and script mode. A number of hands-on examples are given in Python, NumPy and MySQL to gradually explain the methodology to solve different types of problems and handle data. The programming and database related examples as well as the exercises in those chapters are required to be solved in a computer and verified with the given outputs.

© NCERT not to be republished

The chapters in this book have two additional components — activities for self assessment and 'think and reflect' to generate further interest in the learner.

Group projects through case studies are proposed to solve complex problems. Some exercises have been made in case-study form to promote problem-finding and problem-solving skills.

These chapters have been written by involving practicing teachers as well as subject experts. These have been iteratively peer-reviewed. Several iterations have resulted into this book. Thanks to the authors and reviewers for their valuable contribution.

Comments and suggestions are welcome to make this endeavour par excellence.

Dr. Rejaul Karim Barbhuiya
Assistant Professor,
Department of Education in
Science and Mathematics, NCERT

© NCERT not to be republished

Textbook Development Committee

Members

Anuradha Khattar, *Assistant Professor*, Miranda House, University of Delhi, Delhi

Chetna Khanna, *Freelance Educationist*, Delhi

Gurpreet Kaur, *PGT* (Computer Science), GD Goenka Public School, Delhi

Harita Ahuja, *Assistant Professor*, Acharya Narendra Dev College, University of Delhi, Delhi

Mudasir Wani, *Assistant Professor*, Govt. Degree College for Women, Srinagar, Jammu and Kashmir

Om Vikas, *Professor* (Retd.), Formerly Director, ABV-IIITM, Gwalior, Madhya Pradesh

Priti Rai Jain, *Assistant Professor*, Miranda House, University of Delhi, Delhi

Rinku Kumari, *PGT* (Computer Science), Kendriya Vidyalaya, Sainik Vihar, Delhi

Sharanjit Kaur, *Associate Professor*, Acharya Narendra Dev College, University of Delhi, Delhi

Tapasi Ray, *Formerly Global IT Director*, Huntsman Corporation, Singapore

Member-Coordinator

Rejaul Karim Barbhuiya, *Assistant Professor*, DESM, NCERT, Delhi

© NCERT not to be republished

Acknowledgements

The National Council of Educational Research and Training acknowledges the valuable contributions of the individuals and organisation involved in the development of Informatics Practices textbook for Class XI.

The council expresses its gratitude to the syllabus development team including MPS Bhatia, *Professor*, Netaji Subhas Institute of Technology, Delhi; T V Vijay Kumar, *Professor*, School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi; Zahid Raza, *Associate Professor*, School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi; Vipul Shah, *Principal Scientist*, Tata Consultancy Services, and the CSpathshala team; Smruti Ranjan Sarangi, *Associate Professor*, Department of Computer Science and Engineering, Indian Institute of Technology Delhi; Vikram Goyal, *Associate Professor*, Indraprastha Institute of Information Technology (IIIT) Delhi; Vandana Tyagi, *PGT* (Computer Science), Kendriya Vidyalaya, JNU, Delhi and Mamur Ali, *Assistant Professor*, Central Institute of Educational Technology, NCERT, New Delhi.

The council is thankful to the following resource persons for their contribution in editing, reviewing, and refining the manuscript of this book: D.N. Sansanwal, Retd. Professor, Devi Ahilya Vishwavidyalaya, Indore; Veer Saini Dixit, *Assistant Professor*, Atma Ram Sanatan Dharma College, University of Delhi, Delhi; Mukesh Kumar, *Teacher*, DPS RK Puram, Delhi; Gautam Sarkar, *Teacher*, Modern School, Barakhamba Road, Delhi; Aswin K. Dash, *Teacher,* Mother's International School, Delhi; Nancy Sehgal, *Teacher,* Mata Jai Kaur Public School, Delhi; Neelima Gupta, *Professor*, Department of Computer Science, University of Delhi; Anamika Gupta, *Assistant Professor*, Shaheed Sukhdev College of Business Studies, University of Delhi. The council further acknowledges the contribution of Anuja Krishn, *freelance editor*, for refining the chapters from language point of view.

The council in greatful to Dinesh Kumar, *Professor* and *Head*, DESM for his valuable cooperation and support throughout the development of this book.

The council also gratefully acknowledges the contributions of Meetu Sharma, *Graphic Designer;* Kanika Walecha, *DTP Operator*; Pooja, *Junior Project Fellow;* Hari Darshan Lodhi and Junaid Ahmed, *DTP Operator* (Contractual); Chanchal Chauhan, *Proofreader* (Contractual) and Aishwarya Bhattacharyya, *Assistant Editor* (Contractual), in shaping this book. The contributions of the office of the APC, DESM and Publication Division, NCERT, New Delhi, in bringing out this book are also duly acknowledged.

© NCERT not to be republished

CONTENTS

FOREWORD *iii*

PREFACE *v*

CHAPTER 1 COMPUTER SYSTEM **1**

1.1 Introduction to Computer System 1
1.2 Evolution of Computer 3
1.3 Computer Memory 5
1.4 Software 9

CHAPTER 2 EMERGING TRENDS **15**

2.1 Introduction to Emerging Trends 15
2.2 Artificial Intelligence (AI) 16
2.3 Big Data 19
2.4 Internet of Things (IoT) 21
2.5 Cloud Computing 23
2.6 Grid Computing 25
2.7 Blockchains 26

CHAPTER 3 BRIEF OVERVIEW OF PYTHON **31**

3.1 Introduction to Python 31
3.2 Python Keywords 34
3.3 Identifiers 34
3.4 Variables 34
3.5 Data Types 35
3.6 Operators 38
3.7 Expressions 41
3.8 Input and Output 42
3.9 Debugging 43
3.10 Functions 44
3.11 `if..else` Statements 46
3.12 `for` Loop 48
3.13 Nested Loops 50

CHAPTER 4 WORKING WITH LISTS AND DICTIONARIES **55**

4.1 Introduction to List 55
4.2 List Operations 57
4.3 Traversing a List 59
4.4 List Methods and Built-in Functions 60

© NCERT not to be republished

4.5 List Manipulation 62
4.6 Introduction to Dictionaries 67
4.7 Traversing a Dictionary 69
4.8 Dictionary Methods and Built-in Functions 69
4.9 Manipulating Dictionaries 71

CHAPTER 5 UNDERSTANDING DATA 81
5.1 Introduction to Data 81
5.2 Data Collection 85
5.3 Data Storage 86
5.4 Data Processing 87
5.5 Statistical Techniques for Data Processing 88

CHAPTER 6 INTRODUCTION TO NUMPY 95
6.1 Introduction 95
6.2 Array 96
6.3 NumPy Array 96
6.4 Indexing and Slicing 100
6.5 Operations on Arrays 102
6.6 Concatenating Arrays 104
6.7 Reshaping Arrays 105
6.8 Splitting Arrays 106
6.9 Statistical Operations on Arrays 107
6.10 Loading Arrays from Files 109
6.11 Saving NumPy Arrays in Files on Disk 112

CHAPTER 7 DATABASE CONCEPTS 123
7.1 Introduction 123
7.2 File System 124
7.3 Database Management System 127
7.4 Relational Data Model 132
7.5 Keys in a Relational Database 136

CHAPTER 8 INTRODUCTION TO STRUCTURED QUERY LANGUAGE (SQL) 143
8.1 Introduction 143
8.2 Structured Query Language (SQL) 144
8.3 Data Types and Constraints in MySQL 145
8.4 SQL for Data Definition 146
8.5 SQL for Data Manipulation 153
8.6 SQL for Data Query 156
8.7 Data Updation and Deletion 166

© NCERT not to be republished

11149CH01

Chapter 1

Computer System

"A computer would deserve to be called intelligent if it could deceive a human into believing that it was human."

— Alan Turing

In this chapter

- Introduction to Computer System
- Evolution of Computer
- Computer Memory
- Software

1.1 Introduction to Computer System

A computer is an electronic device that can be programmed to accept data (input), process it and generate result (output). A computer along with additional hardware and software together is called a computer system.

A computer system primarily comprises of a central processing unit, memory, input/output devices, and storage devices. All these components function together as a single unit to deliver the desired output. A computer system comes in various forms and sizes. It can vary

© NCERT not to be republished

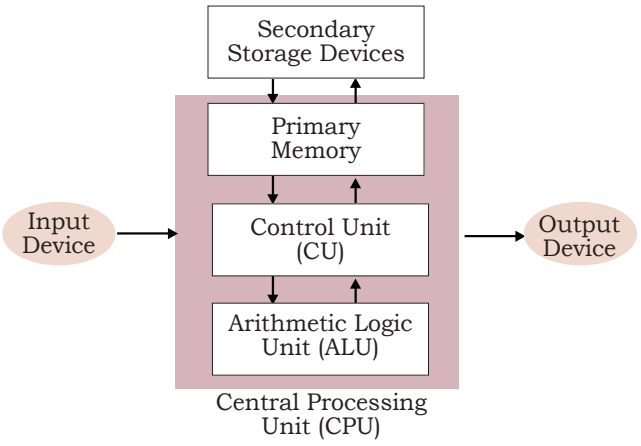

Figure 1.1: Components of a Computer System

from a high-end server to a personal desktop, laptop, tablet computer, or smartphone.

Figure 1.1 shows the block diagram of a computer system. The directed lines represent the flow of data and signal between the components.

1.1.1 Central Processing Unit (CPU)

It is the electronic circuitry of a computer that carries out the actual processing and is usually referred to as the brain of the computer. It is also commonly called 'processor' also. Physically, a CPU can be placed on one or more microchips called integrated circuits (IC). The ICs comprise semiconductor materials.

The CPU is given instructions and data through programs. The CPU then fetches the program and data from the memory and performs arithmetic and logical operations as per the given instructions and stores the result back to memory.

While processing, the CPU stores the data as well as instructions in its local memory, 'called' registers. Registers are part of the CPU chip and they are limited in size and number. Different registers are used for storing data, instructions or intermediate results.

Other than the registers, the CPU has two main components — Arithmetic Logic Unit (ALU) and Control Unit (CU). ALU performs all the arithmetic and logic operations that need to be done as per the instruction in a program. CU controls sequential instruction execution, interprets instructions and guides data flow through the computer's memory, ALU and input or output devices. CPU is also popularly known as microprocessor.

Figure 1.2: Input Devices

1.1.2 Input Devices

The devices through which control signals are sent to a computer are termed as input devices. These devices convert the input data into a digital form that is acceptable by the computer system. Some examples of input devices include keyboard, mouse, scanner, touch screen, etc., as shown in Figure 1.2. Specially designed braille keyboards are also available to help the visually impaired for entering data into a computer. Besides, we can now enter data through voice, for example, we can

© NCERT not to be republished

use Google voice search to search the web where we can input the search string through our voice.

Data entered through input device is temporarily stored in the main memory (also called RAM) of the computer system. For permanent storage and future use, the data as well as instructions are stored permanently in additional storage locations called secondary memory.

1.1.3 Output Devices

The device that receives data from a computer system for display, physical production, etc., is called output device. It converts digital information into human-understandable form. For example, monitor, projector, headphone, speaker, printer, etc. Some output devices are shown in Figure 1.3. A braille display monitor is useful for a visually challenged person to understand the textual output generated by computers.

Figure 1.3: Output Devices

A printer is the most commonly used device to get output in physical (hardcopy) form. Three types of commonly used printers are inkjet, laserjet and dot matrix. Now-a-days, there is a new type of printer called 3D-printer, which is used to build physical replica of a digital 3D design. These printers are being used in manufacturing industries to create prototypes of products. Their usage is also being explored in the medical field, particularly for developing body organs.

1.2 Evolution of Computer

From the simple calculator to a modern day powerful data processor, computing devices have evolved in a relatively short span of time. The evolution of computing devices is shown through a timeline at Figure 1.5.

The Von Neumann architecture is shown in Figure 1.4. It consists of a Central Processing Unit (CPU) for processing arithmetic and logical instructions, a memory to store data and programs, input and output devices and communication channels to send/receive the output data. Electronic Numerical Integrator And Computer (ENIAC) is the first binary programmable computer based on Von Neumann architecture.

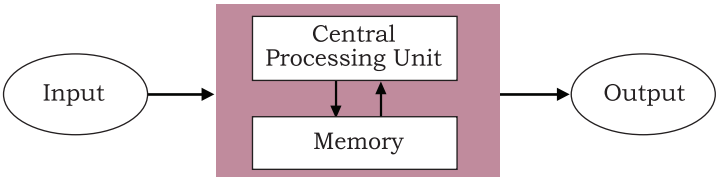

Figure 1.4: Von Neumann Architecture for the Computer

During the 1970s, Large Scale Integration (LSI) of electronic circuits allowed integration of

© NCERT not to be republished

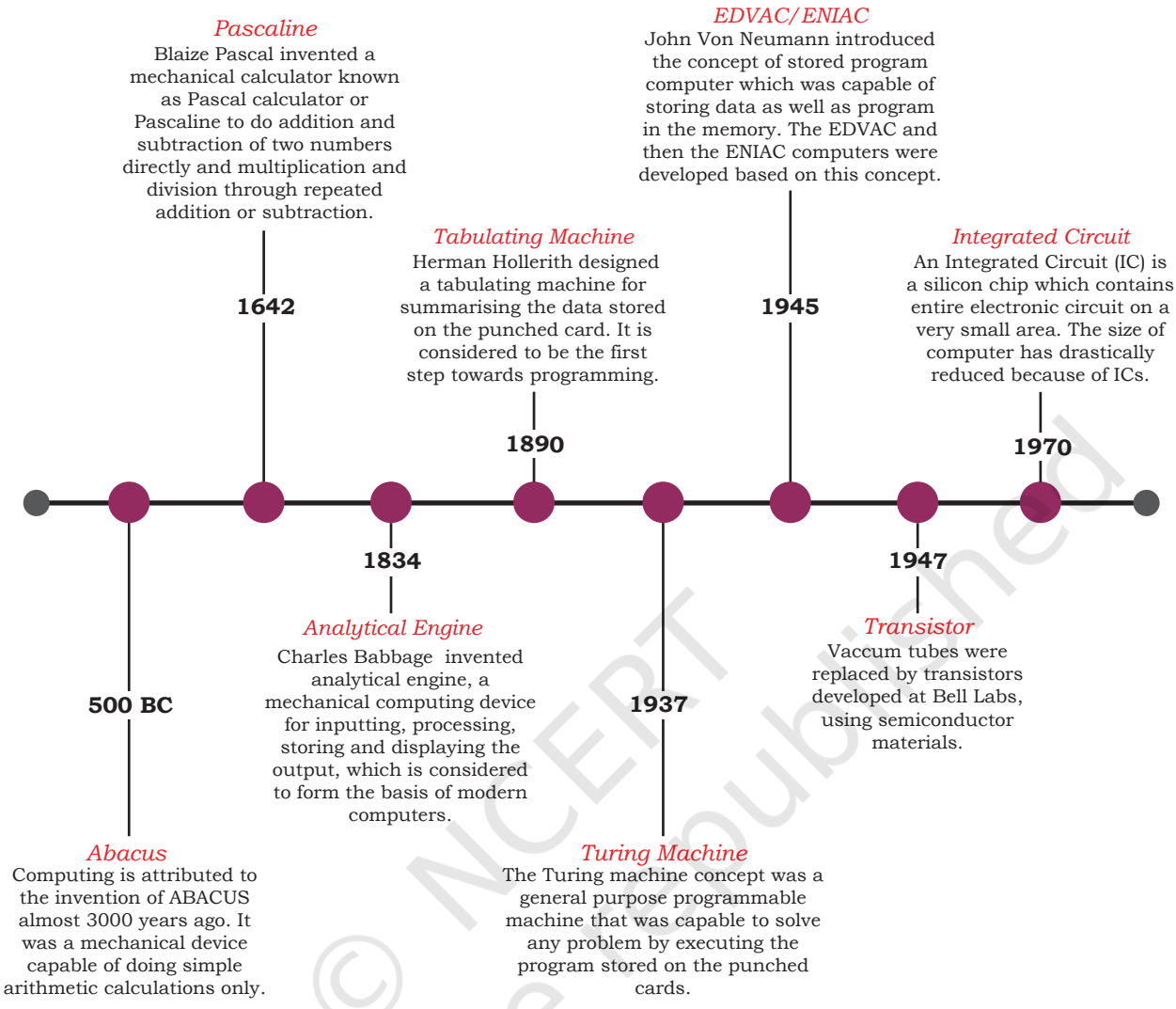

Figure 1.5: Timeline Showing Key Inventions in Computing Technology

complete CPU on a single chip, called microprocessor. Moore's Law predicted exponential growth in number of transistors that could be assembled in a single microchip. In 1980s, the processing power of computers increased exponentially by integrating around 3 million components on a small-sized chip termed as Very Large Scale Integration (VLSI). Further advancement in technology has made it feasible to fabricate high density of transistors and other components (approx 106 components) on a single IC called Super Large Scale Integration (SLSI) as shown in Figure 1.6.

A punched card is a piece of stiff paper that stores digital data in the form of holes at predefined positions.

IBM introduced its first personal computer (PC) for the home user in 1981, Apple introduced Macintosh machines in 1984. The popularity of the PC surged by the introduction of Graphical User Interface (GUI)

© NCERT not to be republished

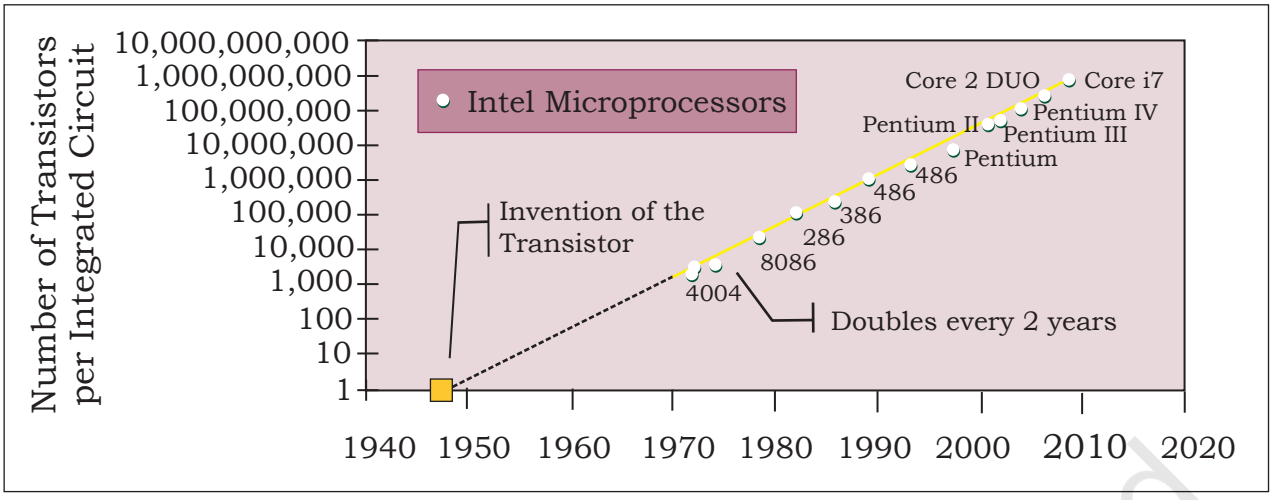

Figure 1.6: Exponential Increase in Number of Transistors used in ICs Over Time

based operating systems by Microsoft and others in place of computers with only command line interface, like UNIX or DOS. Around 1990s, the growth of world wide web (WWW) further accelerated mass usage of computers and thereafter computers have become an indispensable part of everyday life.

Further, with the introduction of laptops, personal computing was made portable to a great extent. This was followed by smartphones, tablets and other personal digital assistants. These devices have leveraged the technological advancements in processor miniaturisation, faster memory, high speed data and connectivity mechanisms.

The next wave of computing devices includes wearable gadgets such as smart watch, lenses, headbands, headphones, etc. Further, smart appliances are becoming a part of the Internet of Things (IoT), by leveraging the power of artificial intelligence.

In 1965, Intel co-founder Gordon Moore introduced Moore's Law which predicted that the number of transistors on a chip would double every two years while the costs would be halved.

1.3 Computer Memory

A computer system needs memory to store the data and instructions for processing. Whenever we talk about the "memory" of a computer system, we usually talk about the main or primary memory. The secondary memory (also called storage device) is used to store data, instructions and results permanently and for future use.

Think and Reflect

How do different components of a computer communicate with each other?

1.3.1 Units of Memory

A computer system uses binary numbers to store and process data. The binary digits 0 and 1, which are the

© NCERT not to be republished

basic units of memory, are called bits. Further, these bits are grouped together to form words. A 4-bit word is called a Nibble. Examples of nibble are 1001, 1010, 0010, etc. A two nibble word, i.e., 8-bit word is called a byte, for example, 01000110, 01111100, 10000001, etc.

Like any other standard unit, bytes are grouped together to make bigger chunks or units of memory. Table 1.1 shows different measurement units for digital data stored in computer memories.

Table 1.1 Measurement units for digital data

Unit	Description	Unit	Description
KB (Kilobyte)	1 KB = 1024 Bytes	PB (Petabyte)	1 PB = 1024 TB
MB (Megabyte)	1 MB = 1024 KB	EB (Exabyte)	1 EB = 1024 PB
GB (Gigabyte)	1 GB = 1024 MB	ZB (Zettabyte)	1 ZB = 1024 EB
TB (Terabyte)	1 TB = 1024 GB	YB (Yottabyte)	1 YB = 1024 ZB

1.3.2 Types of Memory

Human beings memorise many things over a lifetime, and recall from memory to make a decision or take some action. However, we cannot rely on our memory completely, so we make notes and store important data and information using other mediums such as notebook, manual, journal, document etc. for a long-term storage. Similarly, computers have two types of memories namely —primary memory and secondary memory.

(A) Primary Memory

The primary memory is an essential component of a computer system. Program and data are loaded into the primary memory before processing. The CPU interacts directly with the primary memory to perform read/write operation. It is of two types viz. i) Random access memory (RAM), and ii) Read only memory (ROM).

RAM is volatile i.e. as long as the power is supplied to the computer, it retains the data in it. But as soon as the power supply is turned off, all the contents of RAM are wiped out. It is used to store data temporarily while the computer is working. Whenever the computer is started or a software application is launched, the required program and data are loaded into RAM for processing. RAM is usually referred to as main memory and it is faster than the secondary memory or storage devices.

On the other hand, ROM is non-volatile, means its contents are not lost even when the power is turned off. It is used as a small but faster permanent storage for the contents which are rarely changed. For example, the

Think and Reflect

Suppose there is a computer with RAM but no secondary storage. Can we install a software on that computer?

© NCERT not to be republished

startup program (boot loader) that loads the operating system into RAM is stored in a ROM.

(B) Cache Memory

RAM is faster than secondary storage, but not as fast as a computer processor. So, because of RAM, a CPU may have to slow down. To speed up the operations of the CPU, a very high speed memory is placed between the CPU and the primary memory known as cache. It stores the copies of the data from frequently accessed primary memory locations, thus, reducing the average time required to access data from primary memory. When the CPU needs to access memory, it first examines the cache. In case the requirement is met, it is read from the cache, otherwise the primary memory is accessed.

(C) Secondary Memory

Primary memory has limited storage capacity and is either volatile (RAM) or read-only (ROM). Thus, a computer system needs auxiliary or secondary memory to permanently store the data or instructions for future use. The secondary memory is non-volatile and has larger storage capacity than primary memory. It is slower and cheaper than the main memory. But, it cannot be accessed directly by the CPU. Contents of secondary storage need to be first brought into the main memory for the CPU to access. Examples of secondary memory devices include Hard Disk Drive (HDD), CD/DVD, Memory Card, etc., as shown in Figure 1.7.

However, these days, there are secondary storage devices like Solid-State Drive (SSD) which support very fast data transfer speed as compared to earlier HDDs. Also, data transfer between computers have become easier and simpler due to the availability of small sized and portable flash/pen drives.

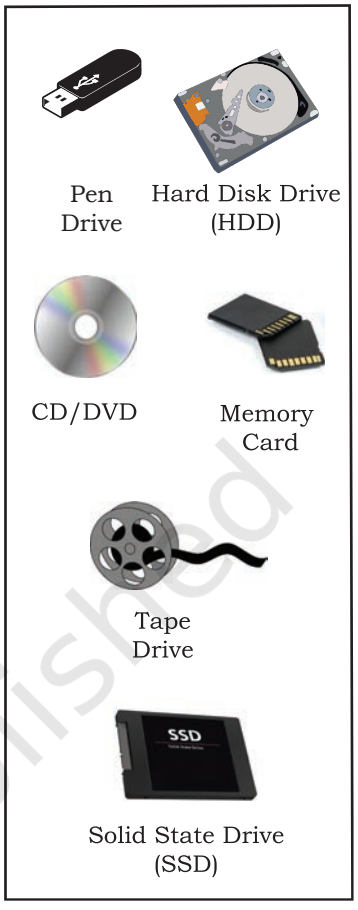

Figure 1.7: Storage Devices

1.3.3 Data Capturing, Storage, and Retrieval

To process the data, we need to first input or capture the data. This is followed by its storage in a file or a database so that it can be used in the future. Whenever data is to be processed, it is first retrieved from the file/database so that we can perform further actions on it.

(A) Data Capturing

It involves the process of gathering data from different sources in digital form. Data may be coptured using, keyboard bar code readers (Used at shopping outlets as

Activity 1.1

List all secondary storage devices available in your school or home.

© NCERT not to be republished

Figure 1.8: Capturing Data using Barcode Reader

shown in Fig. 1.8 (Figure 1.8)), remore sensors on earth orbiting satellites etc. comments/ports over multiple social media are also captured as data. Sometimes, heterogeneity among data sources makes data capturing a complex task.

(B) Data Storage

It is the process of storing the captured data for processing later. Now-a-days data is being produced at a very high rate, and therefore data storage has become a challenging task. However, the decrease in the cost of digital storage devices has helped in simplifying this task. There are numerous digital storage devices available in the market as shown in Figure 1.7.

Data keeps on increasing with time. Hence, the storage devices also require to be upgraded periodically. In large organisations, computers with larger and faster storage called data servers are deployed to store vast amount of data. Such dedicated computers help in processing data efficiently. However, the cost (both hardware and software) of setting up a data server as well as its maintenance is high, especially for small organisations and startups.

Activity 1.2

Visit some of the places like bank, automobile showroom, shopping mall, tehsil office, etc., and find out 2–3 names of tools/instruments used to capture data in digital format.

(C) Data Retrieval

It involves fetching data from the storage devices, for its processing as per the user requirement. As databases grow, the challenges involved in search and retrieval of the data in acceptable time, also increase. Minimising data access time is crucial for faster data processing.

1.3.4 Data Deletion and Recovery

One of the biggest threats associated with digital data is its deletion. The storage devices can malfunction or crash down resulting in the deletion of the stored data. Users can accidentally erase data from storage devices, or a hacker/malware can delete the digital data intentionally.

Deleting digitally stored data means changing the details of data at bit level, which can be very time-consuming. Therefore, when any data is simply deleted, its address entry is marked as free, and that much space is shown as empty to the user, without actually deleting the data.

Activity 1.3

Explore possible ways of recovering deleted data or data from a corrupted device.

In case data gets deleted accidentally or corrupted, there arises a need to recover the data. Recovery of the data is possible only if the contents/memory space

© NCERT not to be republished

marked as deleted have not been overwritten by some other data. Data recovery is a process of retrieving deleted, corrupted and lost data from secondary storage devices.

There are usually two security concerns associated with data. One is its deletion by some unauthorised person or software. These concerns can be avoided by limiting access to the computer system and using passwords for user accounts and files, wherever possible. There is also an option of encrypting files to protect them from unwanted modification.

The other concern is related to unwanted recovery of data by unauthorised user/software. Many a times, we discard our old, broken or malfunctioning storage devices without taking care to delete data. We assume that the contents of deleted files are permanently removed. However, if these storage devices fall into the hands of mischief-mongers, they can easily recover data from such devices; this poses a threat to data confidentiality. This concern can be mitigated by using proper tools to delete or shred data before disposing off any old or faulty storage device.

Activity 1.4

Create a test file and then delete it using Shift+Delete from the keyboard. Now recover the file using the methods you have explored at Activity 1.3.

1.4 SOFTWARE

Till now, we have studied about the physical components or the hardware of the computer system. But the hardware is of no use on its own. Hardware needs to be operated by a set of instructions. These sets of instructions are referred to as software. It is that component of a computer system, which we cannot touch or view physically. It comprises of the instructions and data to be processed using the computer hardware. The computer software and hardware complete any task together.

The software comprises of set of instructions which on execution deliver the desired outcome. In other words, each software is written for some computational purpose. Some examples of software include operating systems like Ubuntu or Windows 7/10, word processing tools like LibreOffice Writer or Microsoft Word, video player like VLC Player, photo editors like Paint and LibreOffice Draw. A document or image stored on the hard disk or pen drive is referred to as a softcopy. Once printed, the document or an image is called a hardcopy.

Hardware refers to the physical components of the computer system which can be seen and touched. For example, RAM, keyboard, printer, monitor, CPU etc. On the other hand, software is a set of instructions and data that makes hardware functional to complete the desired task.

© NCERT not to be republished

1.4.1 Need of Software

The sole purpose of a software is to make computer hardware useful and operational. A software knows how to make different hardware components of a computer work and communicate with each other as well as with the end user. We cannot talk to or instruct the hardware of a computer directly. Hence, software acts as an interface between human users and the hardware.

Depending on the mode of interaction with hardware and functions to be performed, software can be broadly classified into three categories viz. i) System software ii) Programming tools and iii) Application software. The categorisation of software is shown in Figure 1.9.

1.4.2 System Software

The software that provides the basic functionality to operate a computer by interacting directly with its constituent hardware is termed as system software. A system software knows how to operate and use different hardware components of a computer. It provides services directly to the end user, or to some other software. Examples of system software include operating systems, system utilities, device drivers, etc.

(A) Operating System

As the name implies, operating system is a system software that operates the computer. An operating system is the most basic system software, without which other software cannot work. The operating system manages other application programs and provides access and security to the users of the system. Some of the popular operating systems are Windows, Linux, Macintosh, Ubuntu, Fedora, Android, iOS, etc.

(B) System Utilities

Software used for maintenance and configuration of the computer system is called system utility. Some system utilities are shipped with the operating system, for example disk defragmentation tool, formatting utility, system restore utility, etc. Another set of utilities are those which are not shipped with the operating system but are required to improve the performance of the system, for example, anti-virus software, disk cleaner tool, disk compression software, etc.

© NCERT not to be republished

(C) Device Drivers

As the name signifies, the purpose of a device driver is to ensure proper functioning of a particular device. When it comes to the overall working of a computer system, the operating system does the work. But everyday new devices and components are being added to a computer system. It is not possible for operating system alone to manage all of the existing and new peripherals, where each device has diverse characteristics. The responsibility for overall control, operation, and management of a particular device at the hardware level is delegated to its device driver.

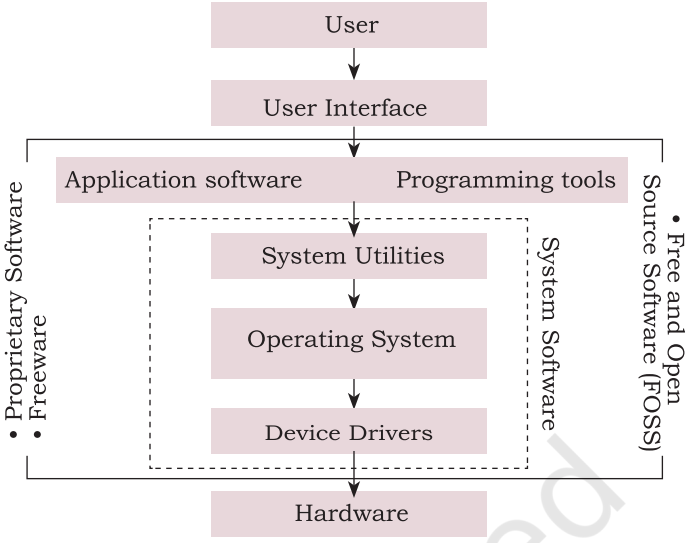

Figure 1.9: Categorisation of Software

The device driver acts as an interface between the device and the operating system. It provides required services by hiding the details of operations performed at the hardware level of the device. Just like a language translator, a device driver acts as a mediator between the operating system and the attached device.

Activity 1.5

Ask your teacher to help you locate any two device drivers installed on your computer.

1.4.3 Application Software

The system software provides the core functionality of the computer system. However, different users need the computer system for different purposes depending upon their requirements. Hence, a new category of software is needed to cater to different requirements of the end-users. This specific software that works on top of the system software is termed as application software. There are again two broad categories of application software: general purpose and customised application software.

(A) General Purpose Software

The application software developed for generic applications, to cater to a bigger audience in general are called general purpose software. Such ready-made application software can be used by end users as per their requirements. For example, spreadsheet tool LibreOffice Calc can be used by any computer user to do calculation or to create an account sheet. Adobe Photoshop, GIMP, Mozilla web browser, iTunes, etc. fall in the category of general purpose software.

A computer system can work without application software, but it cannot work without system software. For example, we can use a computer even if no word processing software is installed, but if no operating system is installed, we can not work on the computer. In other words, the use of computer is possible in the absence of application software.

© NCERT not to be republished

Activity 1.6

With the help of your teacher, install one application software in your computer.

(B) Customised Software

These are custom or tailor-made application software, that are developed to meet the requirements of a specific organisation or an individual. They are better suited to the needs of an individual or an organisation, considering that they are designed as per special requirements. Some examples of user-defined software include websites, school management software, accounting software, etc. It is similar to buying a piece of cloth with specific color and fabric and get it stitched as desired.

1.4.4 Proprietary or Free and Open Source Software

Developers of some software allow public to freely use their software along with source code with an aim to improve further with each other's help. Such software is known as Free and Open Source Software (FOSS). For example, the source code of operating system Ubuntu is freely accessible for anyone with the required knowledge to improve/add new functionality. More examples of FOSS include Python, Libreoffice, Openoffice, Mozilla Firefox, etc. Sometimes, software are freely available for use but source code may not be available. Such software is called freeware. Examples of freeware are Skype, Adobe Reader etc.

When software to be used has to be purchased from the vendor who has the copyright of the software, then it is a proprietary software. Examples of proprietary software include Microsoft Windows, Tally, Quickheal etc. A software can be freeware or open source or proprietary software depending upon the terms and conditions of the person or group who has developed and released that software.

© NCERT not to be republished

SUMMARY

- A computing device, also referred as computer processes the input data as per given instructions to generate desired output.
- Computer processes data to generate information whose further analysis and interpretation yields knowledge.
- Computer system has four physical components viz. i) CPU ii) Primary memory iii) Input device and iv) Output device. They are referred to as hardware of computer.
- Computer system has two types of primary memories viz. i) RAM the volatile memory and ii) ROM the non-volatile memory.
- Software is a set of instructions written to achieve the desired task and are mainly categorised as system software, programming tools and application software.
- Hardware of a computer cannot function on its own. It needs software to be operational or functional.
- Operating system is an interface between the user and the computer and supervises the working of computer system i.e. it monitors and controls the hardware and software of the computer system.

NOTES

© NCERT not to be republished

EXERCISE

1. Name the software required to make a computer functional. Write down its two primary functions.
2. What is the need of RAM? How does it differ from ROM?
3. What is the need for secondary memory?
4. Draw the block diagram of a computer system. Briefly write about the functionality of each component.
5. Differentiate between proprietary software and freeware software. Name two software of each type.
6. Mention any browsers used for browsing the internet.

NOTES

7. Name the input/output device used to do the following:
   a) To output audio
   b) To enter textual data
   c) To make hard copy of a text file
   d) To display the data/information
   e) To enter audio-based command
   f) To build 3D models
   g) To assist a visually impaired individual in entering data
8. Identify the category (system, application, programming tool) of the following software:
   a) Compiler
   b) Assembler
   c) Ubuntu
   d) Text editor
9. Convert the following into bytes:
   a) 2 MB
   b) 3.7 GB
   c) 1.2 TB
10. What is the security threats involved when we throw away electronic gadgets that are non-functional?
11. Write down the type of memory needed to do the following:
   a) To store data permanently
   b) To execute the program
   c) To store the instructions which can not be overwritten.

© NCERT not to be republished

11149CH02

Chapter 2

Emerging Trends

"Computer science is no more about computers than astronomy is about telescopes"

— Edsger Dijkstra

In this chapter

- Introduction to Emerging Trends
- Artificial Intelligence (AI)
- Big Data
- Internet of Things (IoT)
- Cloud Computing
- Grid Computing
- Blockchains

2.1 Introduction to Emerging Trends

Computers have been around for quite some time now. New technologies and initiatives emerge with each passing day. In order to understand the existing technologies and have a better view of the developments around us, we must keep an eye on the emerging trends. Many new technologies are introduced almost every day. Some of these do not succeed and fade away over time. Some of these new technologies prosper and persist over time, gaining attention from users. Emerging trends are the state-of-the-art technologies, which gain

© NCERT not to be republished

popularity and set a new trend among users. In this chapter, we will learn about some emerging trends that will make a huge impact (in the future) on digital economy and interaction in digital societies.

2.2 ARTIFICIAL INTELLIGENCE (AI)

Have you ever wondered how maps in your smartphone are able to guide you to take the fastest route to your destination by analysing real time data, such as traffic congestion? On uploading a photo on a social networking site, has it ever happened that your friends in the photograph were recognised and tagged automatically? These are some of the examples of application of Artificial Intelligence. The intelligent digital personal assistants like Siri, Google Now, Cortana, Alexa are all powered by AI. Artificial Intelligence endeavours to simulate the natural intelligence of human beings into machines, thus making them behave intelligently. An intelligent machine is supposed to imitate some of the cognitive functions of humans like learning, decision-making and problem solving. In order to make machines perform tasks with minimum human intervention, they are programmed to create a knowledge base and make decisions based on it. AI system can also learn from past experiences or outcomes to make new decisions.

A knowledge base is a store of information consisting of facts, assumptions and rules which an AI system can use for decision making.

2.2.1 Machine Learning

Machine Learning is a subsystem of Artificial Intelligence, wherein computers have the ability to learn from data using statistical techniques, without being explicitly programmed by a human being. It comprises algorithms that use data to learn on their own and make predictions. These algorithms, called models, are first trained and tested using a training data and testing data, respectively. After successive trainings, once these models are able to give results to an acceptable level of accuracy, they are used to make predictions about new and unknown data.

Activity 2.1

Find out how NLP is helping differently-abled persons?

2.2.2 Natural Language Processing (NLP)

The predictive typing feature of search engine that helps us by suggesting the next word in the sentence while typing keywords and the spell checking features are examples of Natural Language Processing (NLP). It deals with the interaction between human and

© NCERT not to be republished

computers using human spoken languages, such as Hindi, English, etc.

In fact it is possible to search the web or operate or control our devices using our voice. All this has been possible by NLP. An NLP system can perform text-to-speech and speech-to-text conversion as depicted in Figure 2.1.

Figure 2.1: Use of natural language processing

Machine translation is a rapidly emerging field where machines are able to translate texts from one language to another with fair amount of correctness. Another emerging application area is automated customer service where a computer software can interact with customers to serve their queries or complaints.

2.2.3 Immersive Experiences

With the three-dimensional (3D) videography, the joy of watching movies in theatres has reached to a new level. Video games are also being developed to provide immersive experiences to the player. Immersive experiences allow us to visualise, feel and react by stimulating our senses. It enhances our interaction and involvement, making them more realistic and engaging. Immersive experiences have been used in the field of training, such as driving simulators (Figure 2.2), flight simulator and so on. Immersive experience can be achieved using virtual reality and augmented reality.

Figure 2.2: Driving Simulator

(A) Virtual Reality

Everything that we experience in our reality is perceived through our senses. From this came the idea that if we can present our senses with made-up or non-real information, our perception of reality would also alter in response to that. Virtual Reality (VR) is a three-dimensional, computer-generated situation that simulates the real world. The user can interact with and explore that environment by getting immersed in it while interacting with the objects and other actions of the user. At present, it is achieved with the help of VR Headsets. In order to make the experience of VR more realistic, it promotes other sensory information like sound, smell, motion, temperature, etc. It is a comparatively new field

Figure 2.3: VR Headset

© NCERT not to be republished

Unlike Virtual Reality, the Augmented Reality does not create something new, it just alters or augments the perception of the underlying physical world through additional information.

and has found its applications in gaming (Figure 2.3), military training, medical procedures, entertainment, social science and psychology, engineering and other areas where simulation is needed for a better understanding and learning.

(B) Augmented Reality

The superimposition of computer generated perceptual information over the existing physical surroundings is called as Augmented Reality (AR). It adds components of the digital world to the physical world, along with the associated tactile and other sensory requirements, thereby making the environment interactive and digitally manipulable. Users can access information about the nearest places with reference to their current location. They can get information about places and choose on the basis of user reviews. With thet help of location-based AR App, travellers can access real-time information of historical places just by pointing their camera viewfinder to subjects as depicted in Figure 2.4. Location-based AR apps are major forms of AR apps.

Figure 2.4: Location based Augmented Reality

2.2.4 Robotics

A robot is basically a machine capable of carrying out one or more tasks automatically with accuracy and precision. Unlike other machines, a robot is programmable, which means it can follow the instructions given through computer programs. Robots were initially conceptualised for doing repetitive industrial tasks that are boring or stressful for humans or were labour-intensive. Sensors are one of the prime components of a robot. Robot can be of many types, such as wheeled robots, legged robots, manipulators and humanoids. Robots that resemble humans are known as humanoids. Robots are being used in industries, medical science, bionics, scientific research, military, etc. Some examples are:

- NASA's Mars Exploration Rover (MER) mission is a robotic space mission to study about the planet Mars (Figure 2.5).
- Sophia is a humanoid that uses artificial intelligence, visual data processing, facial recognition and also imitates human gestures and facial expressions, as shown in Figure 2.6.
- A drone is an unmanned aircraft which can be remotely controlled or can fly autonomously through

Activity 2.2

Find out what role are robots playing in the medical field?

Robotics is an interdisciplinary branch of technology requiring applications of mechanical engineering, electronics, and computer science, among others. Robotics is primarily concerned with the design, fabrication, operation, and application of robots.

© NCERT not to be republished

Figure 2.5: NASA's Mars Exploration Rover (MER)

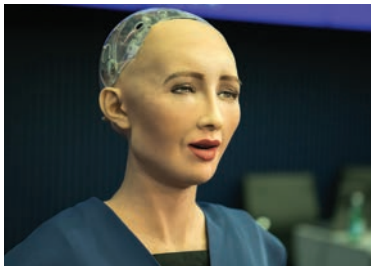

Figure 2.6: Sophia : a Humanoid

Figure 2.7: an unmanned aircraft

software-controlled flight plans in their embedded systems, working in conjunction with onboard sensors and GPS (Figure 2.7). Drones are being used in many fields, such as journalism, filming and aerial photography, shipping or delivery at short distances, disaster management, search and rescue operations, healthcare, geographic mapping and structural safety inspections, agriculture, wildlife monitoring or pooching, besides law-enforcement and border patrolling.

Think and Reflect

Can a drone be helpful in the event of a natural calamity?

2.3 BIG DATA

With technology making an inroad into almost every sphere of our lives, data is being produced at a colossal rate. Today, there are over a billion Internet users, and a majority of the world's web traffic is coming from smartphones. Figure 2.8 shows that at the current pace, around 2.5 quintillion bytes of data are created each day, and the pace is increasing with the continuous evolution of the Internet of Things (IoT).

This results in the generation of data sets of enormous volume and complexity called *Big Data*. Such data cannot be processed and analysed using traditional data

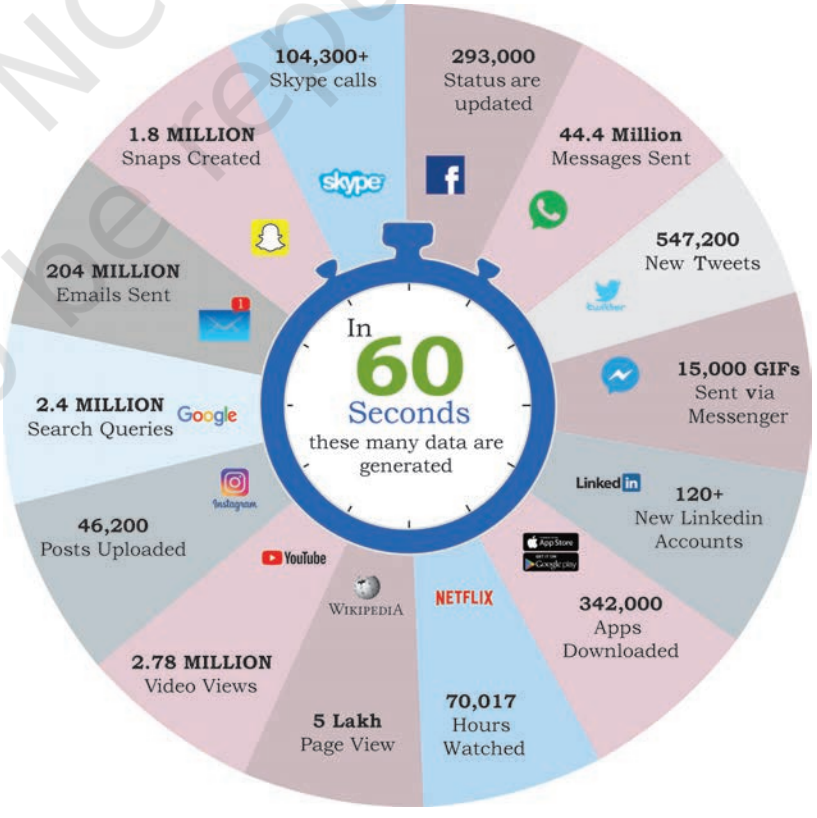

Figure 2.8: Sources of big data (numbers are approximate)

© NCERT not to be republished

Think and Reflect

How are your digital activities contributing to generation of Big data?

processing tools as the data is not only voluminous, but also unstructured like our posts, instant messages and chats, photographs that we share through various sites, our tweets, blog articles, news items, opinion polls and their comments, audio/video chats, etc. Big data not only represents voluminous data, it also involves various challenges like integration, storage, analysis, searching, processing, transfer, querying and visualisation of such data. Big data sometimes hold rich information and knowledge which is of high business value, and therefore there is a keen effort in developing software and methods to process and analyse big data.

2.3.1 Characteristics of Big Data

Big data exhibits following five characteristics shown in Figure 2.9, that distinguish it from traditional data.

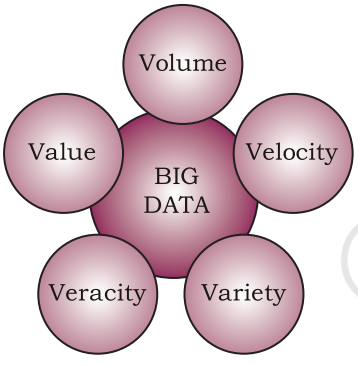

Figure 2.9: Characteristics of big data

(A) Volume

The most prominent characteristic of big data is its enormous size. If a particular data set is of such large size that it is difficult to process it with traditional DBMS tools, it can be termed as big data.

(B) Velocity

It represents the rate at which the data under consideration is being generated and stored. Big data has an exponentially higher rate of generation than traditional data sets.

(C) Variety

It asserts that a data set has varied data, such as structured, semi-structured and unstructured data. Some examples are text, images, videos, web pages and so on.

(D) Veracity

Big data can be sometimes inconsistent, biased, noisy or there can be abnormality in the data or issues with the data collection methods. Veracity refers to the trustworthiness of the data because processing such incorrect data can give wrong results or mislead the interpretations.

(E) Value

Big data is not only just a big pile of data, but also possess to have hidden patterns and useful knowledge which can be of high business value. But as there is cost of investment of resources in processing big data, we should make a preliminary enquiry to see the potential

© NCERT not to be republished

of the big data in terms of value discovery or else our efforts could be in vain.

2.3.2 Data Analytics

Data analytics is the process of examining data sets in order to draw conclusions about the information they contain, with the aid of specialised systems and software.

Data analytics technologies and techniques are becoming popular day-by-day. They are used in commercial industries to enable organisations to make more informed business decisions. In the field of science and technology, it can be useful for researchers to verify or disprove scientific models, theories and hypotheses. Pandas is a library of the programming language Python that can be used as a tool to make data analysis much simpler.

2.4 INTERNET OF THINGS (IoT)

The term computer network that we commonly use is the network of computers. Such a network consists of a laptop, desktop, server, or a portable device like tablet, smartphone, smartwatch, etc., connected through wire or wireless. We can communicate between these devices using Internet or LAN. Now imagine what if our bulbs, fans and refrigerator also became a part of this network. How will they communicate with each other, and what will they communicate? Think about the advantages and tasks that can be accomplished if all these devices with smart connectivity features are able to communicate amongst themselves and we are also able to communicate with them using computers or smartphones!

Activity 2.3

Explore and list a few IoT devices available in the market.

The 'Internet of Things' is a network of devices that have an embedded hardware and software to communicate (connect and exchange data) with other devices on the same network as shown in Figure 2.10. At present, in a typical household, many devices have advanced hardware (microcontrollers) and software. These devices are used

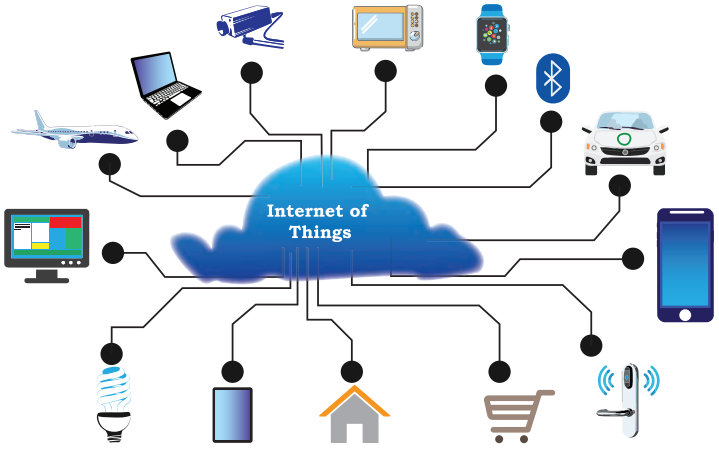

Figure 2.10: Internet of Things (IoT)

© NCERT not to be republished

in isolation from each other, with maximum human intervention needed for operational directions and input data. IoT tends to bring together these devices to work in collaboration and assist each other in creating an intelligent network of things. For example, if a microwave oven, an air conditioner, door lock, CCTV camera or other such devices are enabled to connect to the Internet, we can access and remotely control them on-the-go using our smartphone.

2.4.1 Web of Things (WoT)

Internet of Things allows us to interact with different devices through Internet with the help of smartphones or computers, thus creating a personal network. But to interact with 'n' number of different devices, we need to install 'n' different apps. Wouldn't it be convenient to have one interface to connect all the devices? The web is already being used as a system to communicate with each other. So, will it be possible to use the web in such a way that all things can communicate with each other in the most efficient manner by integrating them together? Web of Things (WoT) allows the use of web services to connect anything in the physical world, besides human identities on web. It will pave way for creating smart homes, smart offices, smart cities and so on.

Activity 2.4

We use GPS to navigate outdoors. VPS is another emerging trend that uses Augmented Reality. Explore and find its other utilities.

2.4.2 Sensors

What happens when you hold your mobile vertically or horizontally? The display also changes to vertical or horizontal with respect to the way we hold our mobile. This is possible with the help of two sensors, namely accelerometer and gyroscope (gyro). The accelerometer sensor in the mobile phones detects the orientation of the phone. The gyroscope sensors tracks rotation or twist of your hand and add to the information supplied by the accelerometer.

Sensors are very commonly used for monitoring and observing elements in real world applications. The evolution of smart electronic sensors is contributing in a large way to the evolution of IoT. It will lead to creation of new sensor-based, intelligent systems.

A smart sensor is a device that takes input from the physical environment and uses built-in computing resources to perform predefined functions upon detection of specific input and then process data before passing it on.

© NCERT not to be republished

2.4.3 Smart Cities

Think and Reflect

What are your ideas of transforming your city into a smart city?

With rapid urbanisation, the load on our cities is increasing day-by-day, and there are challenges in management of resources like land water, waste, air pollution, health and sanitation, traffic congestions, public safety and security, besides the overall city infrastructures including road, rail, bridge, electricity, subways, disaster management, sports facilities, etc. These challenges are forcing many city planners around the world to look for smarter ways to manage them and make cities sustainable and livable.

The idea of a smart city as shown in Figure 2.11 makes use of computer and communication technology along with IoT, WoT to manage and distribute resources efficiently. The smart building shown here uses sensors to detect earthquake tremors and then warn nearby buildings so that they can prepare themselves accordingly. The smart bridge uses wireless sensors to detect any loose bolt, cable or crack. It alerts concerned authorities through SMS. The smart tunnel also uses wireless sensors to detect any leakage or congestion in the tunnel. This information can be sent as wireless signals across the network of sensor nodes to a centralised computer for further analysis.

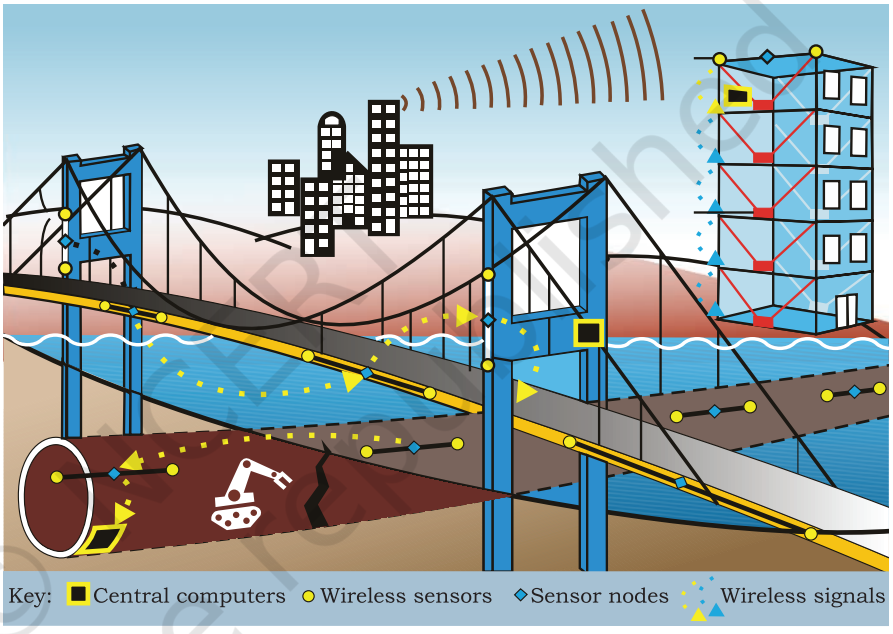

Figure 2.11: Smart City

Every sphere of life in a city like transportation systems, power plants, water supply networks, waste management, law enforcement, information systems, schools, libraries, hospitals and other community services work in unison to optimise the efficiency of city operations and services.

2.5 Cloud Computing

Cloud computing is an emerging trend in the field of information technology, where computer-based services are delivered over the Internet or the cloud, for the case

© NCERT not to be republished

of their accessibility form any where using any smart device. The services comprise software, hardware (servers), databases, storage, etc. These resources are provided by companies called cloud service providers and usually charge on pay per use basis, like the way we pay for electricity usage. We already use cloud services while storing our pictures and files as backup on Internet, or host a website on the Internet. Through cloud computing, a user can run a bigger application or process a large amount of data without having the required storage or processing power on their personal computer as long as they are connected to the Internet. Besides other numerous features, cloud computing offers cost-effective, on-demand resources. A user can avail need-based resources from the cloud at a very reasonable cost.

2.5.1 Cloud Services

A better way to understand the cloud is to interpret everything as a service. A service corresponds to any facility provided by the cloud. There are three standard models to categorise different computing services delivered through cloud as shown in Figure 2.12. These are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

(A) Infrastructure as a Service (IaaS)

The IaaS providers can offer different kinds of computing infrastructure, such as servers, virtual machines (VM), storage and backup facility, network components, operating systems or any other hardware or software. Using IaaS from the cloud, a user can use the hardware infrastructure located at a remote location to configure, deploy and execute any software application on that cloud infrastructure. They can outsource the hardware and software on demand basis and pay as per the usage, thereby they can save the cost of software, hardware and other infrastructures as well as the cost of setting up, maintenance and security.

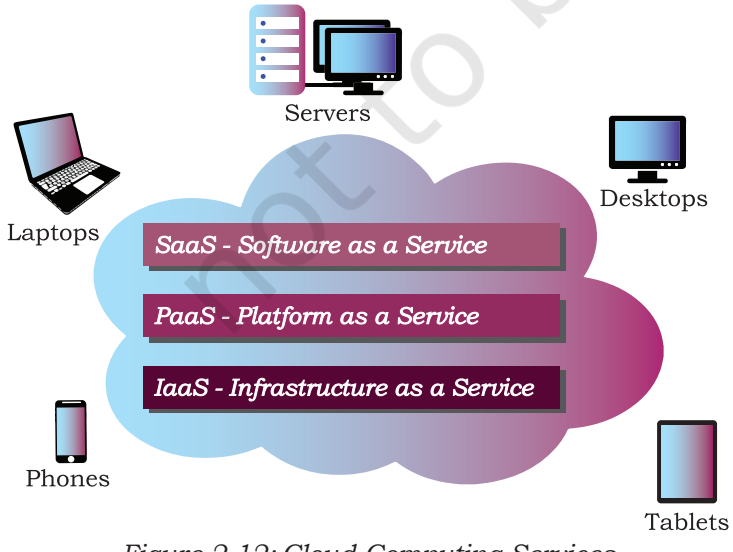

Figure 2.12: Cloud Computing Services

© NCERT not to be republished

(B) Platform as a Service (PaaS)

Through this service, a user can install and execute an application without worrying about the underlying infrastructure and their setup. That is, PaaS provides a platform or environment to develop, test, and deliver software applications. Suppose we have developed a web application using MySQL and Python. To run this application online, we can avail a pre-configured Apache server from cloud having MySQL and Python pre-installed. Thus, we are not required to install MySQL and Python on the cloud, nor do we need to configure the web server (Apache, nginx). In PaaS, the user has complete control over the deployed application and its configuration. It provides a deployment environment for developers at a much reduced cost lessening the complexity of buying and managing the underlying hardware and software.

Activity 2.5

Name a few data centers in India along with the major services that they provide.

(C) Software as a Service (SaaS)

SaaS provides on-demand access to application software, usually requiring a licensing or subscription by the user. While using Google doc, Microsoft Office 365, Drop Box, etc., to edit a document online, we use SaaS from cloud. A user is not concerned about installation or configuration of the software application as long as the required software is accessible. Like PaaS, a user is provided access to the required configuration settings of the application software, that they are using at present.

In all of the above standard service models, a user can use on-demand infrastructure or platform or software and is usually charged as per the usage, thereby eliminating the need of a huge investment upfront for a new or evolving organisation. In order to utilise and harness the benefits of cloud computing, Government of India has embarked upon an ambitious initiative — 'GI Cloud' which has been named as 'MeghRaj' (https://cloud.gov.in).

2.6 GRID COMPUTING

A grid is a computer network of geographically dispersed and heterogeneous computational resources as shown in Figure 2.13. Unlike cloud, whose primary focus is to provide services, a grid is more application specific and creates a sense of a virtual supercomputer

© NCERT not to be republished

Think and Reflect

How can some of the emerging trends discussed in this chapter be used as assistive tools for people with disabilities?

with an enormous processing power and storage. The constituent resources are called nodes. These different nodes temporarily come together to solve a single large task and to reach a common goal.

Nowadays, countless computational nodes ranging from hand-held mobile devices to personal computers and workstations are connected to Local Area Network (LAN) or Internet. Therefore, it is economically feasible to reuse or utilise their resources like memory as well as processing power. The grid provides an opportunity to solve computationally intense scientific and research problems without actually procuring a costly hardware.

Grid can be of two types— (i) Data grid, used to manage large and distributed data having the required multi-user access, and (ii) CPU or Processor grid, where processing is moved from one PC to another as needed or a large task is divided into subtasks, and allotted to various nodes for parallel processing.

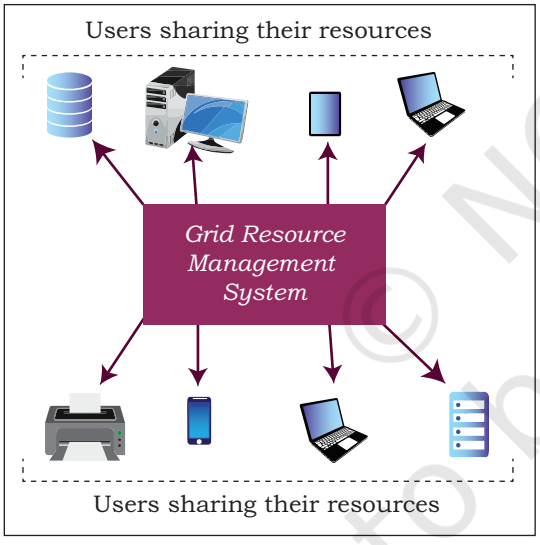

Figure 2.13: Grid computing

Grid computing is different from IaaS cloud service. In case of IaaS cloud service, there is a service provider who rents the required infrastructure to the users. Whereas in grid computing, multiple computing nodes join together to solve a common computational problem.

To set up a grid, by connecting numerous nodes in terms of data as well as CPU, a middleware is required to implement the distributed processor architecture. The Globus toolkit (http://toolkit.globus.org/toolkit) is one such software toolkit used for building grids, and it is as open source. It includes software for security, resource management, data management, communication, fault detection, etc.

2.7 BLOCKCHAINS

Traditionally, we perform digital transactions by storing data in a centralised database and the transactions performed are updated one by one on the database. That is how the ticket booking websites or banks operate. However, since all the data is stored on a central location, there are chances of data being hacked or lost.

The blockchain technology works on the concept of decentralised and shared database where each computer

© NCERT not to be republished

has a copy of the database. A block can be thought as a secured chunk of data or valid transaction. Each block has some data called its header, which is visible to every other node, while only the owner has access to the private data of the block. Such blocks form a chain called blockchain as shown in Figure 2.14. We can define blockchain as a system that allows a group of connected computers to maintain a single updated and secure ledger. Each computer or node that participates in the blockchain receives a full copy of the database. It maintains an 'append only' open ledger which is updated only after all the nodes within the network authenticate the transaction. Safety and security of the transactions are ensured because all the members in the network keep a copy of the blockchain and so it is not possible for a single member of the network to make changes or alter data.

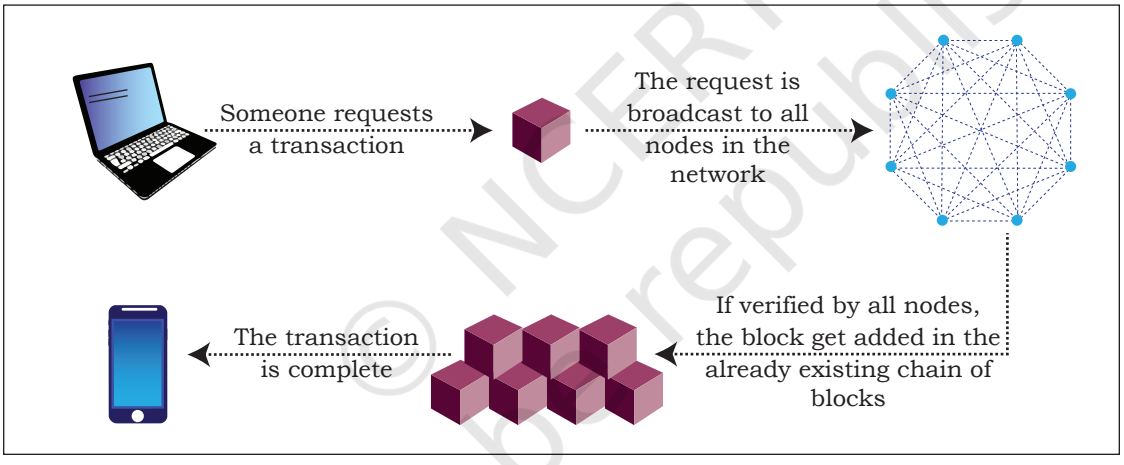

Figure 2.14: Block chain technology

The most popular application of blockchains technology is in digital currency. However, due to its decentralised nature with openness and security, blockchains are being seen as one of the ways to ensure transparency, accountability and efficiency in business as well as in governance systems.

For example, in healthcare, better data sharing between healthcare providers would result in a higher probability of accurate diagnosis, more effective treatments, and the overall increased ability of healthcare organisations to deliver cost-effective care. Another potential application can be for land registration records, to avoid various disputes arising out of land ownership

Think and Reflect

Name any two areas other than those given where the concept of blockchain technology can be useful.

© NCERT not to be republished

NOTES

claims and encroachments. A blockchain based voting system can solve the problem of vote alterations and other issues. Since everything gets stored in the ledger, voting can become more transparent and authentic. The blockchain technology can be used in diverse sectors, such as banking, media, telecom, travel and hospitality and other areas.

SUMMARY

- Artificial Intelligence endeavours to simulate the natural intelligence of human beings into machines thus making them intelligent.
- Machine learning comprises of algorithms that use data to learn on their own and make predictions.
- Natural language processing (NLP) facilitates communicating with intelligent systems using a natural language.
- Virtual reality allows a user to look at, explore, and interact with the virtual surroundings, just like one can do in the real world.
- The superimposition of computer-generated perceptual information over the existing physical surroundings is called augmented reality.
- Robotics can be defined as the science primarily associated with the design, fabrication, theory, and application of robots.
- Big data holds rich information and knowledge which can be of high business value. Five characteristics of big data are: Volume, Velocity, Variety, Veracity, and Value.
- Data analytics is the process of examining data sets in order to draw conclusions about the information they contain.
- The Internet of Things (IoT) is a network of devices that have an embedded hardware and software to communicate (connect and exchange data) with other devices on the same network.
- A sensor is a device that takes input from the physical environment and uses built-in computing resources to perform predefined functions upon detection of specific input and then processes data before passing it on.

© NCERT not to be republished

NOTES

- Cloud computing allows resources located at remote locations to be made available to anyone anywhere. Cloud services can be Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).
- Block chair technology uses a shared data base of chaired blocks where copies of data base exist on multiple computers.

EXERCISE

1. List some of the cloud-based services that you are using at present.
2. What do you understand by the Internet of Things? List some of its potential applications.
3. Write a short note on the following:
   a) Cloud computing
   b) Big data and its characteristics
4. Explain the following along with their applications.
   a) Artificial Intelligence
   b) Machine Learning
5. Differentiate between cloud computing and grid computing with suitable examples.
6. Justify the following statement-
   'Storage of data is cost effective and time saving in cloud computing.'
7. What is on-demand service? How it is provided in cloud computing?
8. Write examples of the following:
   a) Government provided cloud computing platform
   b) Large scale private cloud service providers and the services they provide
9. A company interested in cloud computing is looking for a provider who offers a set of basic services such as virtual server provisioning and on-demand storage that can be combined into a platform for deploying and running customised applications. What type of cloud computing model fits these requirements?
   a) Platform as a Service
   b) Software as a Service
   c) Infrastructure as a Service

© NCERT not to be republished

NOTES

10. Which is not one of the features of IoT devices?
    a) Remotely controllable
    b) Programmable
    c) Can turn themselves off if necessary
    d) All of the above
11. If Government plans to make a smart school by applying IoT concepts, how can each of the following be implemented in order to transform a school into IoT enabled smart school?
    a) e-textbooks
    b) Smart boards
    c) Online tests
    d) Wifi sensors on classrooms doors
    e) Sensors in buses to monitor their location
    f) Wearables (watches or smart belts) for attendance monitoring
12. Five friends plan to try a startup. However, they have a limited budget and limited computer infrastructure. How can they avail the benefits of cloud services to launch their startup?
13. Governments provide various scholarships to students of different classes. Prepare a report on how blockchain technology can be used to promote accountability, transparency and efficiency in distribution of scholarships?
14. How IoT and WoT are related?
15. Match the following:

Column A	**Column B**
You got a reminder to take medication	Smart Parking
You got a sms alert that you forgot to lock the door	Smart Wearable
You got the sms alert that parking space is available near your block	Home Automation
You turned off your LED TV from your wrist watch	Smart Health

© NCERT not to be republished

11149CH03

Chapter 3

Brief Overview of Python

"Don't you hate code that's not properly indented? Making it [indenting] part of the syntax guarantees that all code is properly indented."

— *G. van Rossum*

In this chapter

- Introduction to Python
- Python Keywords
- Identifiers
- Variables
- Data Types
- Operators
- Expressions
- Input and Output
- Debugging
- Functions
- if..else Statements
- for Loop
- Nested Loops

3.1 Introduction to Python

An ordered set of instructions or commands to be executed by a computer is called a *program*. The language used to specify those set of instructions to the computer is called a programming language for example Python, C, C++, Java, etc.

This chapter gives a brief overview of Python programming language. Python is a very popular and easy to learn programming language, created by Guido van Rossum in 1991. It is used in a variety of fields, including software development, web development, scientific computing, big data

© NCERT not to be republished

and Artificial Intelligence. The programs given in this book are written using Python 3.7.0. However, one can install any version of Python 3 to follow the programs given.

> Download Python
>
> The latest version of Python is available on the official website:
>
> https://www.python.org/

3.1.1 Working with Python

To write and run (execute) a Python program, we need to have a Python interpreter installed on our computer or we can use any online Python interpreter. The interpreter is also called *Python shell*. A sample screen of Python interpreter is shown in Figure 3.1. Here, the symbol >>> is called Python prompt, which indicates that the interpreter is ready to receive instructions. We can type commands or statements on this prompt for execution.

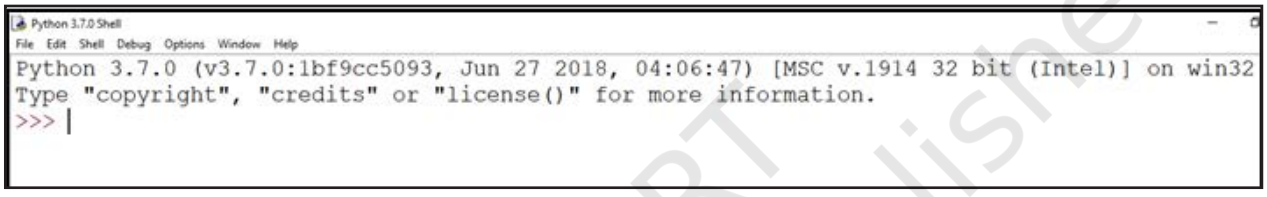

Figure 3.1: Python Interpreter or Shell

3.1.2 Execution Modes

There are two ways to run a program using the Python interpreter:

a) Interactive mode
b) Script mode

(A) Interactive Mode

In the interactive mode, we can type a Python statement on the >>> prompt directly. As soon as we press enter, the interpreter executes the statement and displays the result(s), as shown in Figure 3.2.

Working in the interactive mode is convenient for testing a single line code for instant execution. But in the interactive mode, we cannot save the statements for

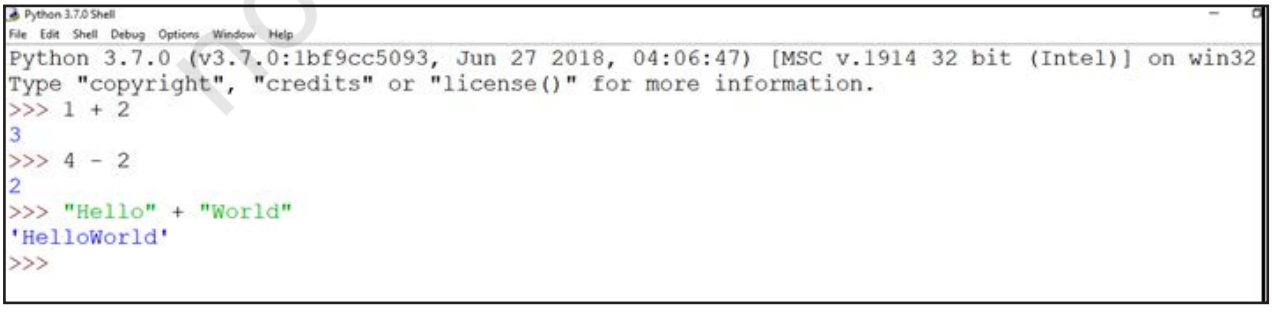

Figure 3.2: Python Interpreter in Interactive Mode

© NCERT not to be republished

future use and we have to retype the statements to run them again.

(B) Script Mode

In the script mode, we can write a Python program in a file, save it and then use the interpreter to execute the program from the file. Such program files have a .py extension and they are also known as scripts. Usually, beginners learn Python in interactive mode, but for programs having more than a few lines, we should always save the code in files for future use. Python scripts can be created using any editor. Python has a *built-in editor* called IDLE which can be used to create programs. After opening the IDLE, we can click File>New File to create a new file, then write our program on that file and save it with a desired name. By default, the Python scripts are saved in the Python installation folder.

IDLE : Integrated Development and Learning Environment

Figure 3.3: Python Code in Script Mode (prog3-1.py)

To execute a Python program in script mode,

a) Open the program using an editor, for example IDLE as shown in Figure 3.3.
b) In IDLE, go to [Run]->[Run Module] to execute the prog3-1.py as shown in Figure 3.4.
c) The output appears on shell as shown in Figure 3.5.

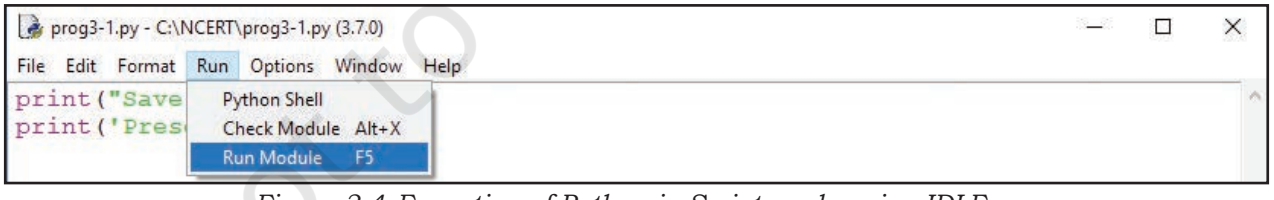

Figure 3.4: Execution of Python in Script mode using IDLE

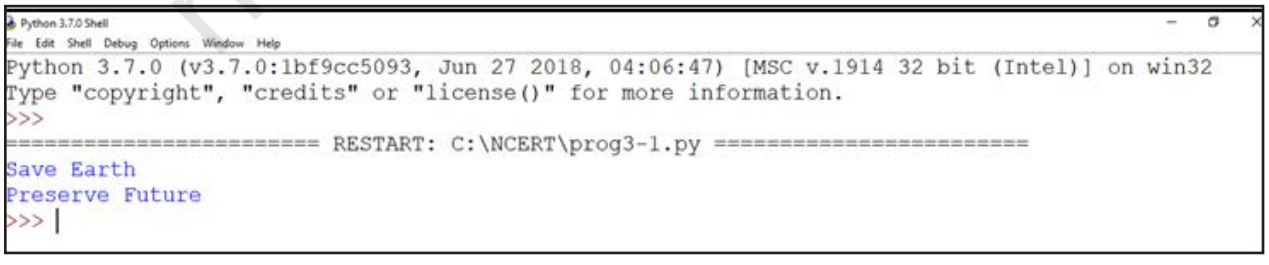

Figure 3.5: Output of a Program prog 3-1.py executed in Script Mode

© NCERT not to be republished

NOTES

3.2 PYTHON KEYWORDS

Keywords are reserved words. Each keyword has a specific meaning to the Python interpreter. As Python is case sensitive, keywords must be written exactly as given in Table 3.1.

Table 3.1 Python keywords

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

3.3 IDENTIFIERS

In programming languages, *identifiers* are names used to identify a variable, function, or other entities in a program. The rules for naming an identifier in Python are as follows:

- The name should begin with an uppercase or a lowercase alphabet or an underscore sign (_). This may be followed by any combination of characters a-z, A-Z, 0-9 or underscore (_). Thus, an identifier cannot start with a digit.
- It can be of any length. (However, it is preferred to keep it short and meaningful).
- It should not be a keyword or reserved word given in Table 3.1.
- We cannot use special symbols like !, @, #, $, %, etc. in identifiers.

For example, to find the average of marks obtained by a student in three subjects namely Maths, English, Informatics Practices (IP), we can choose the identifiers as `marksMaths`, `marksEnglish`, `marksIP` and `avg` rather than a, b, c, or A, B, C, as such alphabets do not give any clue about the data that variable refers to.

```
avg = (marksMaths + marksEnglish + marksIP)/3
```

3.4 VARIABLES

Variable is an identifier whose value can change. For example variable age can have different value for different person. Variable name should be unique in a program. Value of a variable can be string (for example,

© NCERT not to be republished

'b', 'Global Citizen'), number (for example 10,71,80.52) or any combination of alphanumeric (alphabets and numbers for example 'b10') characters. In Python, we can use an assignment statement to create new variables and assign specific values to them.

```
gender   = 'M'
message  = "Keep Smiling"
price    = 987.9
```

Variables must always be assigned values before they are used in the program, otherwise it will lead to an error. Wherever a variable name occurs in the program, the interpreter replaces it with the value of that particular variable.

Comments are used to add a remark or a note in the source code. Comments are not executed by interpreter. They are added with the purpose of making the source code easier for humans to understand. They are used primarily to document the meaning and purpose of source code.

In Python, a single line comment starts with # (hash sign). Everything following the # till the end of that line is treated as a comment and the interpreter simply ignores it while executing the statement.

Program 3-2 Write a Python program to find the sum of two numbers.

```
#Program 3-2
#To find the sum of two given numbers
num1 = 10
num2 = 20
result = num1 + num2
print(result)
#print function in python displays the output
```

Output:

```
30
```

Program 3-3 Write a Python program to find the area of a rectangle given that its length is 10 units and breadth is 20 units.

```
#Program 3-3
#To find the area of a rectangle
length = 10
breadth = 20
area = length * breadth
print(area)
```

Output:

```
200
```

3.5 Data Types

Every value belongs to a specific data type in Python. Data type identifies the type of data which a variable can hold and the operations that can be performed on those data. Figure 3.6 enlists the data types available in Python.

© NCERT not to be republished

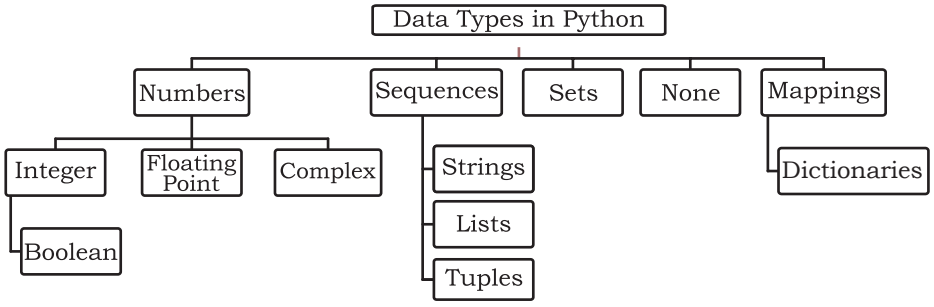

Figure 3.6: Different Data Types in Python

3.5.1 Number

Number data type stores numerical values only. It is further classified into three different types: `int`, `float` and `complex`.

Table 3.2 Numeric data types

Type/ Class	Description	Examples
`int`	integer numbers	-12, -3, 0, 123, 2
`float`	floating point numbers	-2.04, 4.0, 14.23
`complex`	complex numbers	3 + 4i, 2 - 2i

Boolean data type (`bool`) is a subtype of integer. It is a unique data type, consisting of two constants, `True` and `False`. Boolean `True` value is non-zero. Boolean `False` is the value zero.

Let us now try to execute few statements in interactive mode to determine the data type of the variable using built-in function `type()`.

Example 3.1

```
>>> quantity = 10
>>> type(quantity)
<class 'int'>

>>> Price = -1921.9
>>> type(price)
<class 'float'>
```

Variables of simple data types like integer, float, boolean etc. hold single value. But such variables are not useful to hold multiple data values, for example, names of the months in a year, names of students in a class, names and numbers in a phone book or the list of artefacts in a museum. For this, Python provides sequence data types like Strings, Lists, Tuples, and mapping data type Dictionaries.

3.5.2 Sequence

A Python sequence is an ordered collection of items, where each item is indexed by an integer value. Three

© NCERT not to be republished

NOTES

types of sequence data types available in Python are Strings, Lists and Tuples. A brief introduction to these data types is as follows:

(A) String

String is a group of characters. These characters may be alphabets, digits or special characters including spaces. String values are enclosed either in single quotation marks (for example 'Hello') or in double quotation marks (for example "Hello"). The quotes are not a part of the string, they are used to mark the beginning and end of the string for the interpreter. For example,

```
>>> str1 = 'Hello Friend'
>>> str2 = "452"
```

We cannot perform numerical operations on strings, even when the string contains a numeric value. For example `str2` is a numeric string.

(B) List

List is a sequence of items separated by commas and items are enclosed in square brackets [ ]. Note that items may be of different date types.

Example 3.2

```
#To create a list
>>> list1 = [5, 3.4, "New Delhi", "20C", 45]
#print the elements of the list list1
>>> list1
[5, 3.4, 'New Delhi', '20C', 45]
```

(C) Tuple

Tuple is a sequence of items separated by commas and items are enclosed in parenthesis ( ). This is unlike list, where values are enclosed in brackets [ ]. Once created, we cannot change items in the tuple. Similar to List, items may be of different data types.

Example 3.3

```
#create a tuple tuple1
>>> tuple1 = (10, 20, "Apple", 3.4, 'a')
#print the elements of the tuple tuple1
>>> print(tuple1)
(10, 20, "Apple", 3.4, 'a')
```

3.5.3 Mapping

Mapping is an unordered data type in Python. Currently, there is only one standard mapping data type in Python called Dictionary.

© NCERT not to be republished

(A) Dictionary

Dictionary in Python holds data items in key-value pairs and Items are enclosed in curly brackets {}. dictionaries permit faster access to data. Every key is separated from its value using a colon (:) sign. The key value pairs of a dictionary can be accessed using the key. Keys are usually of string type and their values can be of any data type. In order to access any value in the dictionary, we have to specify its key in square brackets [ ].

Example 3.4

```
#create a dictionary
>>> dict1 = {'Fruit':'Apple',
'Climate':'Cold', 'Price(kg)':120}
>>> print(dict1)
{'Fruit': 'Apple', 'Climate': 'Cold',
'Price(kg)': 120}
#getting value by specifying a key
>>> print(dict1['Price(kg)'])
120
```

Python compares two strings lexicographically (According to the theory and practice of composing and writing dictionary), using ASCII value of the characters. If the first character of both strings are same, the second character is compared, and so on.

3.6 OPERATORS

An operator is used to perform specific mathematical or logical operation on values. The values that the operator works on are called *operands*. For example, in the expression `10 + num`, the value `10`, and the variable `num` are operands and the + (plus) sign is an operator. Python supports several kind of operators, their categorisation is briefly explained in this section.

3.6.1 Arithmetic Operators

Python supports arithmetic operators (Table 3.3) to perform the four basic arithmetic operations as well as modular division, floor division and exponentiation.

'+' operator can also be used to concatenate two strings on either side of the operator.

```
>>> str1 = "Hello"
>>> str2 = "India"
>>> str1 + str2
'HelloIndia'
```

'*' operator repeats the item on left side of the operator if first operand is a string and second operand is an integer value.

```
>>> str1 = 'India'
>>> str1 * 2
'IndiaIndia'
```

© NCERT not to be republished

Table 3.3 Arithmetic operators in Python

Operator	Operation	Description	Example (Try in Lab)

+	Addition	Adds two numeric values on either side of the operator	`>>> num1 = 5`
			`>>> num2 = 6`
			`>>> num1 + num2`
			`11`

-	Subtraction	Subtracts the operand on the right from the operand on the left	`>>> num1 = 5`
			`>>> num2 = 6`
			`>>> num1 - num2`
			`-1`

*	Multiplication	Multiplies the two values on both sides of the operator	`>>> num1 = 5`
			`>>> num2 = 6`
			`>>> num1 * num2`
			`30`

/	Division	Divides the operand on the left by the operand on the right of the operator and returns the quotient	`>>> num1 = 5`
			`>>> num2 = 2`
			`>>> num1 / num2`
			`2.5`

%	Modulus	Divides the operand on the left by the operand on the right and returns the remainder	`>>> num1 = 13`
			`>>> num2 = 5`
			`>>> num1 % num2`
			`3`

//	Floor Division	Divides the operand on the left by the operand on the right and returns the quotient by removing the decimal part. It is sometimes also called integer division.	`>>> num1 = 5`
			`>>> num2 = 2`
			`>>> num1 // num2`
			`2`
			`>>> num2 // num1`
			`0`

**	Exponent	Raise the base to the power of the exponent. That is, multiply the base as many times as given in the exponent	`>>> num1 = 3`
			`>>> num2 = 4`
			`>>> num1 ** num2`
			`81`

Operators (+) and (*) work in similar manner for other sequence data types like list and tuples.

3.6.2 Relational Operators

Relational operator compares the values of the operands on its either side and determines the relationship among them. Conside the given Python variables `num1 = 10, num2 = 0, num3 = 10, str1 = "Good", str2 = "Afternoon"` for the following examples in Table 3.4:

Table 3.4 Relational operators in Python

Operator	Operation	Description	Example (Try in Lab)

==	Equals to	If values of two operands are equal, then the condition is True, otherwise it is False.	`>>> num1 == num2`
			`False`
			`>> str1 == str2`
			`False`

© NCERT not to be republished

!=	Not equal to	If values of two operands are not equal, then condition is True, otherwise it is False	`>>> num1 != num2`
			`True`
			`>>> str1 != str2`
			`True`
			`>>> num1 != num3`
			`False`

>	Greater than	If the value of the left operand is greater than the value of the right operand, then condition is True, otherwise it is False.	`>>> num1 > num2`
			`True`
			`>>> str1 > str2`
			`True`

<	Less than	If the value of the left operand is less than the value of the right operand, the condition is true otherwise it is False	`>>> num1 < num3` `False`

Similarly, there are other relational operators like <= and >=.

3.6.3 Assignment Operators

Assignment operator assigns or changes the value of the variable on its left, as shown in Table 3.5.

Table 3.5 Assignment operators in Python

Operator	Description	Example (Try in Lab)

=	Assigns value from right side operand to left side operand	`>>> num1 = 2`
		`>>> num2 = num1`
		`>>> num2`
		`2`
		`>>> country = 'India'`
		`>>> country`
		`'India'`

+=	It adds the value of right side operand to the left side operand and assigns the result to the left side operand.
	Note: x+ = y is same as x = x + y
		`>>> num1 = 10`
		`>>> num2 = 2`
		`>>> num1 += num2`
		`>>> num1`
		`12`
		`>>> num2`
		`2`

-=	It subtracts the value of right side operand from the left side operand and assigns the result to left side operand.
	Note: x– = y is same as x = x – y
		`>>> num1 = 10`
		`>>> num2 = 2`
		`>>> num1 -= num2`
		`>>> num1`
		`8`

Similarly, there are other assignment operators like *=, /=, %=, //=, and **=.

3.6.4 Logical Operators

There are three logical operators (Table 3.6) supported by Python. These operators (`and, or, not`) are to be written in lower case only. The logical operator evaluates to either `True` or `False` based on the logical operands on its either side.

© NCERT not to be republished

Table 3.6 Logical operators in Python

Operator	Operation	Description	Example (Try in Lab)

and	Logical AND	If both operands are True, then condition becomes True	>>> num1 = 10
			>>> num2 = -20
			>>> num1 == 10 and num2 == -20
			True
			>>> num1 == 10 and num2 == 10
			False

or	Logical OR	If any of the two operands are True, then condition becomes True	>>> num1 = 10
			>>> num2 = 2
			>>> num1 >= 10 or num2 >= 10
			True
			>>> num1 <= 5 or num2 >= 10
			False

not	Logical NOT	Used to reverse the logical state of its operand	>>> num1 = 10
			>>> not (num1 == 20)
			True
			>>> not (num1 == 10)
			False

3.6.5 Membership Operators

Membership operator (Table 3.7) is used to check if a value is a member of the given sequence or not.

Table 3.7 Membership operators in Python

Operator	Description	Example (Try in Lab)

in	Returns True if the variable or value is found in the specified sequence and False otherwise	>>> numSeq = [1,2,3]
		>>> 2 in numSeq
		True
		>>> '1' in numSeq
		False
		#'1' is a string while
		#numSeq contains number 1.

not in	Returns True if the variable/value is not found in the specified sequence and False otherwise	>>> numSeq = [1,2,3]
		>>> 10 not in numSeq
		True
		>>> 1 not in numSeq
		False

3.7 EXPRESSIONS

An expression is defined as a combination of constants, variables and operators. An expression always evaluates to a value. A value or a standalone variable is also considered as an expression but a standalone operator is not an expression. Some examples of valid expressions are given below.

```
(i) num - 20.4          (iii) 23/3 -5 * 7(14 -2)
(ii) 3.0 + 3.14          (iv) "Global"+"Citizen"
```

© NCERT not to be republished

NOTES

3.7.1 Precedence of Operators

So far we have seen different operators and examples of their usage. When an expression contains more than one operator, their precedence (order or hierarchy) determines which operator should be applied first. Higher precedence operator is evaluated before the lower precedence operator. In the following example, '*' and '/' have higher precedence than '+' and '-'.

Note:

a) Parenthesis can be used to override the precedence of operators. The expression within () is evaluated first.

b) For operators with equal precedence, the expression is evaluated from left to right.

Example 3.5 How will Python evaluate the following expression?

```
20 + 30 * 40
```

Solution:

```
#precedence of * is more than that of +
= 20 + 1200                    #Step 1
= 1220                         #Step 2
```

Example 3.6 How will Python evaluate the following expression?

```
(20 + 30) * 40
```

Solution:

```
= (20 + 30) * 40               # Step 1
#using parenthesis(), we have forced
precedence of + to be more than that of *
= 50 * 40                      # Step 2
= 2000                         # Step 3
```

Example 3.7 How will the following expression be evaluated?

```
15.0 / 4.0 + (8 + 3.0)
```

Solution:

```
= 15.0 / 4.0 + (8.0 + 3.0)     #Step 1
= 15.0 / 4.0 + 11.0            #Step 2
= 3.75 + 11.0                  #Step 3
= 14.75                        #Step 4
```

3.8 INPUT AND OUTPUT

Sometimes, we need to enter data or enter choices into a program. In Python, we have the `input()` function for taking values entered by input device such as a keyboard. The `input()` function prompts user to enter data. It accepts all user input (whether alphabets,

© NCERT not to be republished

NOTES

numbers or special character) as string. The syntax for `input()` is:

```
variable = input([Prompt])
```

`Prompt` is the string we may like to display on the screen prior to taking the input, but it is optional. The `input()` takes exactly what is typed from the keyboard, converts it into a string and assigns it to the variable on left hand side of the assignment operator (=).

Example 3.8

```
>>> fname = input("Enter your first name: ")
Enter your first name: Arnab
>>> age = input("Enter your age: ")
Enter your age: 19
```

The variable `fname` gets the string 'Arnab' as input. Similarly, the variable `age` gets '19' as string. We can change the datatype of the string data accepted from user to an appropriate numeric value. For example, the `int()` function will convert the accepted string to an integer. If the entered string is non-numeric, an error will be generated.

Example 3.9

```
#function int() to convert string to integer
>>> age = int(input("Enter your age: "))
Enter your age: 19
>>> type(age)
<class 'int'>
```

Python uses the `print()` function to output data to standard output device — the screen. The function `print()` evaluates the expression before displaying it on the screen. The syntax for `print()` is:

```
print(value)
```

Example 3.10

Statement	Output
`print("Hello")`	`Hello`
`print(10*2.5)`	`25.0`

3.9 DEBUGGING

Due to errors, a program may not execute or may generate wrong output. :

i) Syntax errors

ii) Logical errors

iii) Runtime errors

© NCERT not to be republished

NOTES

3.9.1 Syntax Errors

Like any programming language, Python has rules that determine how a program is to be written. This is called syntax. The interpreter can interpret a statement of a program only if it is syntactically correct. For example, parentheses must be in pairs, so the expression (10 + 12) is syntactically correct, whereas (7 + 11 is not due to absence of right parenthesis. If any syntax error is present, the interpreter shows error message(s) and stops the execution there. Such errors need to be removed before execution of the program.

3.9.2 Logical Errors

A logical error/bug (called semantic error) does not stop execution but the program behaves incorrectly and produces undesired /wrong output. Since the program interprets successfully even when logical errors are present in it, it is sometimes difficult to identify these errors.

For example, if we wish to find the average of two numbers 10 and 12 and we write the code as 10 + 12/2, it would run successfully and produce the result 16, which is wrong. The correct code to find the average should have been (10 + 12) /2 to get the output as 11.

3.9.3 Runtime Error

A runtime error causes abnormal termination of program while it is executing. Runtime error is when the statement is correct syntactically, but the interpreter can not execute it.

For example, we have a statement having division operation in the program. By mistake, if the denominator value is zero then it will give a runtime error like “division by zero”.

The process of identifying and removing logical errors and runtime errors is called debugging. We need to debug a program so that is can run successfully and generate the desired output.

3.10 FUNCTIONS

A function refers to a set of statements or instructions grouped under a name that perform specified tasks. For repeated or routine tasks, we define a function. A function is defined once and can be reused at multiple

© NCERT not to be republished

NOTES

places in a program by simply writing the function name, i.e., by calling that function.

Suppose we have a program which requires to calculate compound interest at multiple places. Now instead of writing the formula to calculate the interest every time, we can create a function called CalcCompInt and inside that function we write the code to take inputs (like interest rate, duration, principle), calculate interest, and display output. We can simply call the function by writing the function name CalcCompInt whenever compound interest is to be computed and thus reuse the code to save time and efforts.

Python has many predefined functions called built-in functions. We have already used two built-in functions `print()` and `input()`. A module is a python file in which multiple functions are grouped together. These functions can be easily used in a Python program by importing the module using `import` command. Use of built-in functions makes programming faster and efficient. To use a built-in function we must know the following about that function:

- Function Name — name of the function.
- Arguments — While calling a function, we may pass value(s), called argument, enclosed in parenthesis, to the function. The function works based on these values. A function may or may not have argument(s).
- Return Value – A function may or may not return one or more values. A function performs operations on the basis of argument (s) passed to it and the result is passed back to the calling point. Some functions do not return any value.

Let us consider the following Python program using three built-in functions `input()`, `int()` and `print()`:

```
#Calculate square of a number
num = int(input("Enter the first number"))
square = num * num
print("the square of", num, " is ", square)
```

Observe:

- Two built-in functions are used in the first statement, `int()` and `input()`. The third line has a function `print()`.
- The input function accepts an argument, “Enter your name”. Argument(s) is the value(s) passed within the parenthesis.

© NCERT not to be republished

NOTES

- Similarly the print function has four arguments "the square of", num, "is", square separated by commas.
- The int function in the first line takes as argument the value entered by the user from the keyboard and converts it into a string and returns it. Thus the return value from the int() function is an integer.

Some of the most commonly used built-in functions in Python are listed in Table 3.8 under four broad categories.

Table 3.8 Some commonly used built-in functions in Python

Input/ Output	Datatype Conversion	Mathematical Functions	Other Functions
input()	bool()	abs()	__import__()
print()	chr()	divmod()	len()
	dict()	max()	range()
	float()	min()	type()
	int()	pow()	
	list()	sum()	
	ord()		
	set()		
	str()		
	tuple()		

3.11 if..else STATEMENTS

Usually statements in a program are executed one after another. However, there are situations when we have more than one option to choose from, based on the outcome of certain conditions. This can be done using if..else conditional statements. Conditional statements let us write program to do different tasks or take different paths based on the outcome of the conditions.

There are three ways to write if..else statements:

- if statement — executes the statement(s) inside if when the condition is true. ‘

Example 3.11

```
age = int(input("Enter your age "))
if age >= 18: # use ‘:’ to indicate end of condition.
    print("Eligible to vote")
```

© NCERT not to be republished

- `if...else` statement executes the statement(s) inside `if` when the condition is true, otherwise executes the statement(s) inside `else` (when the condition is false)

```
#Program to subtract smaller number from the
#larger number and display the difference.
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
if num1 > num2:
     diff = num1 - num2
else:
     diff = num2 - num1
print("The difference of",num1,"and",num2,
"is",diff)
```

Output:

```
Enter first number: 5
Enter second number: 6
The difference of 5 and 6 is 1
```

- `if...elif....else` is use dot check multiple conditions and execute statements accordingly. Meaning of `elif` is `elseif`. We can also write `elseif` instead of `elif` for more clarity.

Example 3.12 Check whether a number is positive, negative, or zero.

```
number = int(input("Enter a number: ")
if number > 0:
     print("Number is positive")
elif number < 0:
     print("Number is negative")
else:
     print("Number is zero")
```

When the conditional statements appear, the Python interpreter executes code inside one block that is selected based on the condition. Number of `elif` is dependent on the number of conditions to be checked. If the first condition is false, then the next condition is checked, and so on. If one of the conditions is true, then the corresponding indented block executes, and the `if` statement terminates. After that, the statements outside the `if..else` are executed or the program terminates if there are no further statements.

Python uses indentation for block as well as for nested block structures. Leading whitespace (spaces and tabs) at the beginning of a statement is called indentation. In Python, the same level of indentation associates statements into a single block of code. The interpreter checks indentation levels very strictly and throws up syntax errors if indentation is not correct. It is a common practice to use a single tab for each level of indentation.

© NCERT not to be republished

3.12 For Loop

Sometimes we need to repeat certain things for a particular number of times. For example, a program has to display attendance for every student of a class. Here the program has to execute the print statement `for` every student. In programming, this kind of repetition is called looping or iteration, and it is done using for statement. The `for` statement is used to iterate over a range of values or a sequence. The loop is executed for each item in the range. The values can be numeric, string, list, or tuple.

When all the items in the range are exhausted, the statements within loop are not executed and Python interpreter starts executing the statements immediately following the `for` loop. While using `for` loop, we should know in advance the number of times the loop will execute.

Syntax of the for Loop:

```
for <control-variable> in <sequence/items in
range>:
    <statements inside body of the
     loop>
```

Program 3-4 Program to print even numbers in a given sequence using `for` loop.

```
#Program 3-4
#Print even numbers in the given sequence
numbers = [1,2,3,4,5,6,7,8,9,10]
for num in numbers:
     if (num % 2) == 0:
          print(num,'is an even Number')
```

Output:

```
2  is an even Number
4  is an even Number
6  is an even Number
8  is an even Number
10 is an even Number
```

Note: Body of the loop is indented with respect to the `for` statement.

© NCERT not to be republished

3.12.1 The range() Function

The `range()` is a built-in function in Python. Syntax of `range()` function is:

```
range([start], stop[, step])
```

It is used to create a list containing a sequence of integers from the given `start` value upto `stop` value (excluding `stop` value), with a difference of the given `step` value. If start value is not specified, by default the list starts from 0. If step is also not specified, by default the value is incremented by 1 in each iteration. All parameters of range() function must be integers. The step parameter can be a positive or a negative integer excluding zero.

Example 3.13

```
>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
#start value is given as 2
>>>  list(range(2, 10))
[2, 3, 4, 5, 6, 7, 8, 9]
#step value is 5 and start value is 0
>>>  list(range(0, 30, 5))
[0, 5, 10, 15, 20, 25]
#step value is -1. Hence, decreasing
#sequence is generated
>>>  list(range(0, -9, -1))
[0, -1, -2, -3, -4, -5, -6, -7, -8]
```

The function range() is often used in `for` loops for generating a sequence of numbers.

Program 3-5 Program to print the multiples of 10 for numbers in a given range.

```
#Program 3-5
#Print multiples of 10 for numbers in a range
for num in range(5):
     if num > 0:
          print(num * 10)
```

Output:

```
10
20
30
40
```

© NCERT not to be republished

3.13 NESTED LOOPS

A loop may contain another loop inside it. A loop inside another loop is called a nested loop.

Program 3-6 Program to demonstrate working of nested for loops.

```
#Program 3-6
#Demonstrate working of nested for loops
for var1 in range(3):
    print( "Iteration " + str(var1 + 1) + " of outer loop")
    for var2 in range(2):    #nested loop
        print(var2 + 1)
    print("Out of inner loop")
print("Out of outer loop")
```

Output:

```
Iteration 1 of outer loop
1
2
Out of inner loop
Iteration 2 of outer loop
1
2
Out of inner loop
Iteration 3 of outer loop
1
2
Out of inner loop
Out of outer loop
```

© NCERT not to be republished

Summary

- Python is an open-source, high level, interpreter-based language that can be used for a multitude of scientific and non-scientific computing purposes.
- Comments are non-executable statements in a program.
- An identifier is a user defined name given to a variable or a constant in a program.
- Process of identifying and removing errors from a computer program is called debugging.
- Trying to use a variable that has not been assigned a value gives an error.
- There are several data types in Python — integer, boolean, float, complex, string, list, tuple, sets, None and dictionary.
- Operators are constructs that manipulate the value of operands. Operators may be unary or binary.
- An expression is a combination of values, variables, and operators.
- Python has input() function for taking user input.
- Python has print() function to output data to a standard output device.
- The `if` statement is used for decision making.
- Looping allows sections of code to be executed repeatedly under some condition.
- `for` statement can be used to iterate over a range of values or a sequence.
- The statements within the body of `for` loop are executed till the range of values is exhausted.

Notes

© NCERT not to be republished

Exercise

1. Which of the following identifier names are invalid and why?

a)	`Serial_no.`	e)	`Total_Marks`
b)	`1st_Room`	f)	`total-Marks`
c)	`Hundred$`	g)	`_Percentage`
d)	`Total Marks`	h)	`True`

NOTES

2. Write the corresponding Python assignment statements:
   a) Assign 10 to variable length and 20 to variable breadth.
   b) Assign the average of values of variables length and breadth to a variable sum.
   c) Assign a list containing strings 'Paper', 'Gel Pen', and 'Eraser' to a variable stationery.
   d) Assign the strings 'Mohandas', 'Karamchand', and 'Gandhi' to variables first, middle and last.
   e) Assign the concatenated value of string variables first, middle and last to variable fullname. Make sure to incorporate blank spaces appropriately between different parts of names.

3. Which data type will be used to represent the following data values and why?
   a) Number of months in a year
   b) Resident of Delhi or not
   c) Mobile number
   d) Pocket money
   e) Volume of a sphere
   f) Perimeter of a square
   g) Name of the student
   h) Address of the student

4. Give the output of the following when `num1 = 4, num2 = 3, num3 = 2`
   a) num1 += num2 + num3
   b) print (num1)
   c) num1 = num1 ** (num2 + num3)
   d) print (num1)
   e) num1 **= num2 + c
   f) num1 = '5' + '5'
   g) print(num1)
   h) print(4.00/(2.0+2.0))
   i) num1 = 2+9*((3*12)-8)/10
   j) print(num1)
   k) num1 = float(10)
   l) print (num1)
   m) num1 = int('3.14')

© NCERT not to be republished

n) print (num1)

o) print(10 != 9 and 20 >= 20)

p) print(5 % 10 + 10 < 50 and 29 <= 29)

5. Categorise the following as syntax error, logical error or runtime error:

   a) 25 / 0

   b) num1 = 25; num2 = 0; num1/num2

6. Write a Python program to calculate the amount payable if money has been lent on simple interest. Principal or money lent = P, Rate = R% per annum and Time = T years. Then Simple Interest (SI) = (P x R x T)/ 100.

   Amount payable = Principal + SI.

   P, R and T are given as input to the program.

7. Write a program to repeat the string "GOOD MORNING" n times. Here n is an integer entered by the user.

8. Write a program to find the average of 3 numbers.

9. Write a program that asks the user to enter one's name and age. Print out a message addressed to the user that tells the user the year in which he/she will turn 100 years old.

10. What is the difference between `else` and `elif` construct of `if` statement?

11. Find the output of the following program segments:

a)
```
for i in range(20,30,2):
    print(i)
```

b)
```
country = 'INDIA'
for  i in country:
    print (i)
```

c)
```
i = 0; sum = 0
while i < 9:
        if i % 4 == 0:
            sum = sum + i
        i = i + 2
print (sum)
```

Case Study Based Question

Schools use "Student Management Information System" (SMIS) to manage student related data. This system provides facilities for:

© NCERT not to be republished

NOTES

Name of School

Student Name: PQR Roll No: 99
Class: XI Section: A
Address : Address Line 1
Address Line 2

City: ABC Pin Code: 999999
Parent's/ Guardian's Contact No: 9999999999

- Recording and maintaining personal details of students.
- Maintaining marks scored in assessments and computing results of students.
- Keeping track of student attendance, and
- Managing many other student-related data in the school. Let us automate the same step by step.

Identify the personal details of students from your school identity card and write a program to accept these details for all students of your school and display them in this format.

© NCERT not to be republished

11149CH04

Chapter 4

Working with Lists and Dictionaries

"Computer Science is a science of abstraction – creating the right model for a problem and devising the appropriate mechanizable techniques to solve it."

— A. Aho and J. Ullman

In this chapter

- » Introduction to List
- » List Operations
- » Traversing a List
- » List Methods and Built-in Functions
- » List Manipulation
- » Introduction to Dictionaries
- » Traversing a Dictionary
- » Dictionary Methods and Built-in Functions
- » Manipulating Dictionaries

4.1 Introduction to List

The data type list is an ordered sequence which is mutable and made up of one or more elements. Unlike a string which consists of only characters, a list can have elements of different data types such as integer, float, string, tuple or even another list. A list is very useful to group elements of mixed data types. Elements of a list are enclosed in square brackets and are separated by comma.

Example 4.1

```
#list1 is the list of six even numbers
>>> list1 = [2,4,6,8,10,12]
>>> print(list1)
[2, 4, 6, 8, 10, 12]
```

© NCERT not to be republished

NOTES

```
#list2 is the list of vowels
>>> list2 = ['a','e','i','o','u']
>>> print(list2)
['a', 'e', 'i', 'o', 'u']

#list3 is the list of mixed data types
>>> list3 = [100,23.5,'Hello']
>>> print(list3)
[100, 23.5, 'Hello']

#list4 is the list of lists called nested
#list
>>> list4 =[['Physics',101],['Chemistry',202],
            ['Mathematics',303]]
>>> print(list4)
[['Physics', 101], ['Chemistry', 202],
 ['Mathematics', 303]]
```

4.1.1 Accessing Elements in a List

Each element in list is accessed using value called index. The fist index value is 0, the second index is 1 and so on. Elements in the list are assigned index values in increasing order sterling from 0.

To access an element, use square brackets with the index [] value of that element. We may also use negative index value to access elements starting from the last element in the list, having index value -0.

```
#initialing a list named list1
>>> list1 = [2,4,6,8,10,12]
>>> list1[0] #returns first element of list1
2
>>> list1[3] #returns fourth element of list1
8
#Out of range index value for the list returns error
>>> list1[15]

IndexError: list index out of range
#an expression resulting in an integer index
>>> list1[1+4]
12
>>> list1[-1] #return first element from right
12
#length of the list1 is assigned to n
>>> n = len(list1)
>>> print(n)
6
#Get the last element of the list1
>>> list1[n-1]
12
```

© NCERT not to be republished

```
#Get the first element of list1
>>> list1[-n]
2
```

4.1.2 Lists are Mutable

In Python, lists are mutable. It means that the contents of the list can be changed after it has been created.

```
#List list1 of colors
  >>> list1 = ['Red','Green','Blue','Orange']
#change/override the fourth element of list1
  >>> list1[3] = 'Black'
  >>> list1  #print the modified list list1
['Red', 'Green', 'Blue', 'Black']
```

4.2 LIST OPERATIONS

The data type list allows manipulation of its contents through various operations as shown below.

4.2.1 Concatenation

Python allows us to join two or more lists using concatenation operator using symbol +.

Concatenation is the merging of two or more values. Example: we can concatenate strings together.

```
#list1 is list of first five odd integers
>>> list1 = [1,3,5,7,9]
#list2 is list of first five even integers
>>> list2 = [2,4,6,8,10]
#Get elements of list1 followed by list2
>>> list1 + list2
[1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
>>> list3 = ['Red','Green','Blue']
>>> list4 = ['Cyan', 'Magenta', 'Yellow'
,'Black']
>>> list3 + list4
['Red','Green','Blue','Cyan','Magenta',
 'Yellow','Black']
```

Note that, there is no change in original lists i.e., `list1, list2, list3, list4` remain the same after concatenation operation. If we want to use the result of two concatenated lists, we should use an assignment operator.

For example,

```
#Join list 2 at the end of list
>>> new List = list 1 + list 2
[1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
```

>> new list The concatenation operator '+' requires that the operands should be of list type only. If we try to concatenate a list with elements of some other data type, `TypeError` occurs.

© NCERT not to be republished

NOTES

```
>>> list1 = [1,2,3]
>>> str1 = "abc"
>>> list1 + str1
TypeError: can only concatenate list (not
"str") to list
```

4.2.2 Repetition

Python allows us to replicate the contents of a list using repetition operator depicted by symbol *.

```
>>> list1 = ['Hello']
#elements of list1 repeated 4 times
>>> list1 * 4
['Hello', 'Hello', 'Hello', 'Hello']
```

4.2.3 Membership

The membership operator `in` checks if the element is present in the list and returns `True`, else returns `False`.

```
>>> list1 = ['Red','Green','Blue']
>>> 'Green' in list1
True
>>> 'Cyan' in list1
False
```

The Operator `not in` transpose returns `True` if the element is not present in the list, else it returns `False`.

```
>>> list1 = ['Red','Green','Blue']
>>> 'Cyan' not in list1
True
>>> 'Green' not in list1
False
```

4.2.4 Slicing

Slicing operations allow us to create new list by taking out elements from an existing list.

```
>>> list1 =['Red','Green','Blue','Cyan',
'Magenta','Yellow','Black']
#subject from indexes 2 to 5 of list 1
>>> list1[2:6]
['Blue', 'Cyan', 'Magenta', 'Yellow']

 #list1 is truncated to the end of the list
>>> list1[2:20] #second index is out of range
['Blue', 'Cyan', 'Magenta', 'Yellow',
'Black']

>>> list1[7:2]      #first index > second index
[]                  #results in an empty list
```

© NCERT not to be republished

```
#return sublist from index 0 to 4
>>> list1[:5]      #first index missing
['Red','Green','Blue','Cyan','Magenta']

#slicing with a given step size
>>> list1[0:6:2]
['Red','Blue','Magenta']
#negative indexes
#elements at index -6,-5,-4,-3 are sliced
>>> list1[-6:-2]
['Green','Blue','Cyan','Magenta']


#both first and last index missing
>>> list1[::2]    #step size 2 on entire list
['Red','Blue','Magenta','Black']

#Access list in the reverse order using
negative step size
>>> list1[::-1]
['Black','Yellow','Magenta','Cyan','Blue',
'Green','Red']
```

4.3 TRAVERSING A LIST

We can access each element of the list or traverse a list using a `for` loop or a `while` loop.

(A) List traversal using `for` loop:

```
>>> list1 = ['Red','Green','Blue','Yellow',
             'Black']
>>> for item in list1:
        print(item)
```

Output:

```
Red
Green
Blue
Yellow
Black
```

Another way of accessing the elements of the list is using range() and len() functions:

```
>>> for i in range(len(list1)):
        print(list1[i])
```

Output:

```
Red
Green
Blue
Yellow
Black
```

`len (list1)` returns the length or total number of elements of list1.
range(n) returns a sequence of numbers starting from 0, increases by 1 and ends at n-1 (one number less than the specified number i.e. is)

© NCERT not to be republished

4.4 LIST METHODS AND BUILT-IN FUNCTIONS

The data type list has several built-in methods that are useful in programming. Some of them are listed in Table 4.1.

Table 4.1 Built-in functions for list manipulation

Method	Description	Example

len()	Returns the length of the list passed as the argument	`>>> list1 = [10,20,30,40,50]`
		`>>> len(list1)`
		`5`

list()	Creates an empty list if no argument is passed	`>>> list1 = list()`
		`>>> list1`
		`[ ]`

	Creates a list if a sequence is passed as an argument	`>>> str1= 'aeiou'`
		`>>> list1 = list(str1)`
		`>>> list1`
		`['a', 'e', 'i', 'o', 'u']`

append()	Appends a single element passed as an argument at the end of the list
	
	A list can also be appended as an element to an existing list
		`>>> list1 = [10,20,30,40]`
		`>>> list1.append(50)`
		`>>> list1`
		`[10, 20, 30, 40, 50]`
		`>>> list1 = [10,20,30,40]`
		`>>> list1.append([50,60])`
		`>>> list1`
		`[10, 20, 30, 40, [50, 60]]`

extend()	Appends each element of the list passed as argument at the end of the given list	`>>> list1 = [10,20,30]`
		`>>> list2 = [40,50]`
		`>>> list1.extend(list2)`
		`>>> list1`
		`[10, 20, 30, 40, 50]`

insert()	Inserts an element at a particular index in the list	`>>> list1 = [10,20,30,40,50]`
		`#inserts element 25 at index value 2`
		`>>> list1.insert(2,25)`
		`>>> list1`
		`[10, 20, 25, 30, 40, 50]`
		`>>> list1.insert(0,100)`
		`>>> list1`
		`[100, 10, 20, 25, 30, 40, 50]`

© NCERT not to be republished

count()	Returns the number of times a given element appears in the list	`>>> list1 = [10,20,30,10,40,10]`
		`>>> list1.count(10)`
		`3`
		`>>> list1.count(90)`
		`0`

find()	Returns index of the first occurrence of the element in the list. If the element is not present, ValueError is generated	`>>> list1 = [10,20,30,20,40,10]`
		`>>> list1.index(20)`
		`1`
		`>>> list1.index(90)`
		`ValueError: 90 is not in list`

remove()	Removes the given element from the list. If the element is present multiple times, only the first occurrence is removed. If the element is not present, then ValueError is generated	`>>> list1 = [10,20,30,40,50,30]`
		`>>> list1.remove(30)`
		`>>> list1`
		`[10, 20, 40, 50, 30]`
		`>>> list1.remove(90)`
		`ValueError:list.remove(x):x not in list`

pop()	Returns the element whose index is passed as argument to this function and also removes it from the list. If no argument is given, then it returns and removes the last element of the list	`>>> list1 = [10,20,30,40,50,60]`
		`>>> list1.pop(3)`
		`40`
		`>>> list1`
		`[10, 20, 30, 50, 60]`
		`>>> list1 = [10,20,30,40,50,60]`
		`>>> list1.pop()`
		`60`
		`>>> list1`
		`[10, 20, 30, 40, 50]`

reverse()	Reverses the order of elements in the given list	`>>> list1 = [34,66,12,89,28,99]`
		`>>> list1.reverse()`
		`>>> list1`
		`[ 99, 28, 89, 12, 66, 34]`
		
		`>>> list1 = [ 'Tiger' ,'Zebra' , 'Lion' , 'Cat' ,'Elephant' ,'Dog']`
		`>>> list1.reverse()`
		`>>> list1`
		`['Dog', 'Elephant', 'Cat', 'Lion', 'Zebra', 'Tiger']`

sort()	Sorts the elements of the given list in place	`>>>list1 = ['Tiger','Zebra','Lion', 'Cat', 'Elephant' ,'Dog']`
		`>>> list1.sort()`
		`>>> list1`
		`['Cat', 'Dog', 'Elephant', 'Lion', 'Tiger', 'Zebra']`
		
		`>>> list1 = [34,66,12,89,28,99]`
		`>>> list1.sort(reverse = True)`
		`>>>list1`
		`[99,89,66,34,28,12]`

© NCERT not to be republished

sorted()	It takes a list as parameter and creates a new list consisting of the same elements but arranged in ascending order	>>>list1 = [23,45,11,67,85,56]
		>>> list2 = sorted(list1)
		>>> list1
		[23, 45, 11, 67, 85, 56]
		>>> list2
		[11, 23, 45, 56, 67, 85]

min()
 	
max()
 	
sum()
	Returns minimum or smallest element of the list
	
	Returns maximum or largest element of the list
	Returns sum of the elements of the list
		>>> list1 = [34,12,63,39,92,44]
		>>> min(list1)
		12
		>>> max(list1)
		92
		>>> sum(list1)
		284

4.5 LIST MANIPULATION

In this chapter, we have learnt to create a list and the different ways to manipulate lists. In the following programs, we will apply the various list manipulation methods.

Program 4-1 Write a program to allow user to perform any those list operation given in a menu. The menu is:

1. Append an element
2. Insert an element
3. Append a list to the given list
4. Modify an existing element
5. Delete an existing element from its position
6. Delete an existing element with a given value
7. Sort the list in the ascending order
8. Sort the list in descending order
9. Display the list.

```
#Program 4-1
#Menu driven program to do various list operations
myList = [22,4,16,38,13] #myList having 5 elements
choice = 0
For attempt in range (3): print ("Attempt number:", attempt)
print("The list 'myList' has the following elements", myList)
print("\nL I S T   O P E R A T I O N S")
print(" 1. Append an element")
print(" 2. Insert an element at the desired position")
print(" 3. Append a list to the given list")
print(" 4. Modify an existing element")
print(" 5. Delete an existing element by its position")
print(" 6. Delete an existing element by its value")
```

© NCERT not to be republished

```
print(" 7. Sort the list in ascending order")
print(" 8. Sort the list in descending order")
    print(" 9. Display the list")
    choice = int(input("ENTER YOUR CHOICE (1-9): "))

    #append element
    if choice == 1:
        element = eval(input("Enter the element to be appended: "))
        myList.append(element)
        print("The element has been appended\n")

    #insert an element at desired position
    elif choice == 2:
        element = eval(input("Enter the element to be inserted: "))
        pos = int(input("Enter the position:"))
        myList.insert(pos,element)
        print("The element has been inserted\n")

    #append a list to the given list
    elif choice == 3:
        newList = eval(input("Enter the list to be appended: "))
        myList.extend(newList)
        print("The list has been appended\n")

    #modify an existing element
    elif choice == 4:
        i = int(input("Enter the position of the element to be
modified: "))
        if i < len(myList):
            newElement = eval(input("Enter the new element: "))
            oldElement = myList[i]
            myList[i] = newElement
            print("The element",oldElement,"has been modified\n")
        else:
            print("Position of the element is more then the length
of list")

    #delete an existing element by position
    elif choice == 5:
        i = int(input("Enter the position of the element to be
deleted: "))
        if i < len(myList):
            element = myList.pop(i)
            print("The element",element,"has been deleted\n")
        else:
            print("\nPosition of the element is more then the length
of list")
```

© NCERT not to be republished

```
        #delete an existing element by value
        elif choice == 6:
            element = int(input("\nEnter the element to be deleted: "))
            if element in myList:
                myList.remove(element)
                print("\nThe element",element,"has been deleted\n")
            else:
                print("\nElement",element,"is not present in the list")

        #list in sorted order
        elif choice == 7:
            myList.sort()
            print("\nThe list has been sorted")

        #list in reverse sorted order
        elif choice == 8:
            myList.sort(reverse = True)
            print("\nThe list has been sorted in reverse order")

        #display the list
        elif choice == 9:
            print("\nThe list is:", myList)
            else:
            print("Choice is not valid")
```

Output:

```
The list 'myList' has the following elements [22, 4, 16, 38, 13]
Attempt number : 1
L I S T   O P E R A T I O N S
  1. Append an element
  2. Insert an element at the desired position
  3. Append a list to the given list
  4. Modify an existing element
  5. Delete an existing element by its position
  6. Delete an existing element by its value
  7. Sort the list in ascending order
  8. Sort the list in descending order
  9. Display the list
ENTER YOUR CHOICE (1-10): 8

The list has been sorted in reverse order
The list 'myList' has the following elements [38, 22, 16, 13, 4]
Attempt number : 2
L I S T   O P E R A T I O N S
  1. Append an element
```

© NCERT not to be republished

```
  2. Insert an element at the desired position
  3. Append a list to the given list
  4. Modify an existing element
  5. Delete an existing element by its position
  6. Delete an existing element by its value
  7. Sort the list in ascending order
  8. Sort the list in descending order
  9. Display the list
ENTER YOUR CHOICE (1-9)5
Enter the position of the element to be deleted: 2
The element 16 has been deleted

The list 'myList' has the following elements [38, 22, 13, 4]
Attempt number : 3
L I S T   O P E R A T I O N S
 1. Append an element
 2. Insert an element at the desired position
 3. Append a list to the given list
 4. Modify an existing element
 5. Delete an existing element by its position
 6. Delete an existing element by its value
 7. Sort the list in ascending order
 8. Sort the list in descending order
 9. Display the list
ENTER YOUR CHOICE (1-9) 10
choice is not valid
```

Program 4-2 A program to calculate average marks of n students where n is entered by the user.

```
#Program 4-2
#create an empty list
list1 = []
print("How many students marks you want to enter: ")
n = int(input())
for i in range(0,n):
     print("Enter marks of student",(i+1),":")
     marks = int(input())
     #append marks in the list
     list1.append(marks)
     #initialize total
     total = 0
     for marks in list1:
```

© NCERT not to be republished

```
            #add marks to total
             total = total + marks
      average = total / n
print("Average marks of",n,"students is:",average)
```

Output:

```
How many students marks you want to enter:
5
Enter marks of student 1:
45
Enter marks of student 2:
89
Enter marks of student 3:
79
Enter marks of student 4:
76
Enter marks of student 5:
55
Average marks of 5 students is: 68.8
```

Program 4-3 Write a program to check if a number is present in the list or not. If the number is present, print the position of the number. Print an appropriate message if the number is not present in the list.

```
#Program 4-3
list1 = []                          #Create an empty list
print("How many numbers do you want to enter in the list: ")
maximum = int(input())
print("Enter a list of numbers: ")
for i in range(0,maximum):
     n = int(input())
     list1.append(n)                 #append numbers to the list
num = int(input("Enter the number to be searched: "))

position = -1
for i in range (0, lin (list1)
if list1[i] == num:                 #number is present
position = i+1                      #save the position of number
if position == -1 :
     print("Number",num,"is not present in the list")
else:
     print("Number",num,"is present at",position + 1, "position")
```

Output:

```
How many numbers do you want to enter in the list
5
```

© NCERT not to be republished

```
Enter a list of numbers:
23
567
12
89
324
Enter the number to be searched:12
Number 12 is present at 3 position
```

4.6 INTRODUCTION TO DICTIONARIES

The data type *dictionary* falls under mapping. It is a mapping between a *set of keys* and a *set of values*. The key-value pair is called an *item*. A key is separated from its value by a colon(:) and consecutive items are separated by commas. Items in dictionaries are ordered, so we will get back the data in the same order in which we had entered the data initially in the dictionary.

4.6.1 Creating a Dictionary

To create a dictionary, the items entered are separated by commas and enclosed in curly braces. Each item is a key value pair, separated through colon (:). The keys in the dictionary must be unique and should be of any immutable data type i.e. number, string or tuple. The values can be repeated and can be of any data type.

Example 4.2

```
#dict1 is an empty dictionary
>>> dict1 = {}
>>> dict1
{}
#dict3 is the dictionary that maps names of
#the students to marks in percentage
>>> dict3 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
>>> dict3
{'Mohan': 95, 'Ram': 89, 'Suhel': 92,
'Sangeeta': 85}
```

4.6.2 Accessing Items in a Dictionary

We have already seen that the items of a sequence (string, list and tuple) are accessed using a technique called indexing. The items of a dictionary are accessed via the keys rather than via their relative positions or indices. Each key serves as the index and maps to a value.

© NCERT not to be republished

NOTES

The following example shows how a dictionary returns the value corresponding to the given key:

```
>>> dict3 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
>>> dict3['Ram']
89
>>> dict3['Sangeeta']
85
#using unspecified key
>>> dict3['Shyam']
KeyError: 'Shyam'
```

In the above examples the key `'Ram'` always maps to the value 89 and key `'Sangeeta'` always maps to the value 85. So the order of items does not matter. If the key is not present in the dictionary we get KeyError.

4.6.3 Membership Operation

The membership operator `in` checks if the key is present in the dictionary and returns `True`, else it returns `False`.

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
>>> 'Suhel' in dict1
True
```

The `not in` operator returns `True` if the key is not present in the dictionary, else it returns `False`.

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
>>> 'Suhel' not in dict1
False
```

4.6.4 Dictionaries are Mutable

Dictionaries are mutable which implies that the contents of the dictionary can be changed after it has been created.

(A) Adding a new item

We can add a new item to the dictionary as shown in the following example:

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
>>> dict1['Meena'] = 78
>>> dict1
{'Mohan': 95, 'Ram': 89, 'Suhel': 92,
'Sangeeta': 85, 'Meena': 78}
```

© NCERT not to be republished

(B) Modifying an existing item

The existing dictionary can be modified by just overwriting the key-value pair. Example to modify a given item in the dictionary:

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
#Marks of Suhel changed to 93.5
>>> dict1['Suhel'] = 93.5
>>> dict1
{'Mohan': 95, 'Ram': 89, 'Suhel': 93.5,
'Sangeeta': 85}
```

4.7 Traversing a Dictionary

We can access each item of the dictionary or traverse a dictionary using `for` loop.

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
'Sangeeta':85}
```

Method 1:

```
>>> for key in dict1:
        print(key,':',dict1[key])
Mohan: 95
Ram: 89
Suhel: 92
Sangeeta: 85
```

Method 2:

```
>>> for key,value in dict1.items():
        print(key,':',value)
Mohan: 95
Ram: 89
Suhel: 92
Sangeeta: 85
```

4.8 Dictionary Methods and Built-in Functions

Python provides many functions to work on dictionaries. Table 4.2 lists some of the commonly used dictionary methods.

Table 4.2 Built-in functions and methods for dictionary

Method	Description	Example

len()	Returns the length or number of key: value pairs of the dictionary passed as the argument	`>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> len(dict1)`
		`4`

© NCERT not to be republished

dict()	Creates a dictionary from a sequence of key-value pairs	`pair1 = [('Mohan',95),('Ram',89), ('Suhel',92),('Sangeeta',85)]`
		`>>> pair1`
		`[('Mohan', 95), ('Ram', 89), ('Suhel', 92), ('Sangeeta', 85)]`
		`>>> dict1 = dict(pair1)`
		`>>> dict1`
		`{'Mohan': 95, 'Ram': 89, 'Suhel': 92, 'Sangeeta': 85}`

keys()	Returns a list of keys in the dictionary	`>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> dict1.keys()`
		`dict_keys(['Mohan', 'Ram', 'Suhel', 'Sangeeta'])`

values()	Returns a list of values in the dictionary	`>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> dict1.values()`
		`dict_values([95, 89, 92, 85])`

items()	Returns a list of tuples (key — value) pair	`>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> dict1.items()`
		`dict_items([( 'Mohan',  95), ('Ram', 89), ('Suhel', 92), ('Sangeeta', 85)])`

get()	Returns the value corresponding to the key passed as the argument
	
	If the key is not present in the dictionary it will return None
		`>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> dict1.get('Sangeeta')`
		`85`
		`>>> dict1.get('Sohan')`
		`>>>`

update()	appends the key-value pair of the dictionary passed as the argument to the key-value pair of the given dictionary	`>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> dict2 = {'Sohan':79,'Geeta':89}`
		`>>> dict1.update(dict2)`
		`>>> dict1`
		`{'Mohan': 95, 'Ram': 89, 'Suhel': 92, 'Sangeeta': 85, 'Sohan': 79, 'Geeta': 89}`
		`>>> dict2`
		`{'Sohan': 79, 'Geeta': 89}`

clear()	Deletes or clear all the items of the dictionary	`>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> dict1.clear()`
		`>>> dict1`
		`{ }`

© NCERT not to be republished

del()	Deletes the item with the given key
	To delete the dictionary from the memory we write:
	`del Dict_name`
		`>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}`
		`>>> del dict1['Ram']`
		`>>> dict1`
		`{'Mohan':95,'Suhel':92, 'Sangeeta': 85}`
		
		`>>> dict1`
		`NameError: name 'dict1' is not defined`

4.9 MANIPULATING DICTIONARIES

In this chapter, we have learnt how to create a dictionary and apply various methods to manipulate it. The following examples show the application of those manipulation methods on dictionaries.

(a) Create a dictionary 'ODD' of odd numbers between 1 and 10, where the key is the decimal number and the value is the corresponding number in words.

```
>>> ODD = {1:'One',3:'Three',5:'Five',7:'Seven',9:'Nine'}
>>> ODD
{1: 'One', 3: 'Three', 5: 'Five', 7: 'Seven', 9: 'Nine'}
```

(b) Display the keys in dictionary 'ODD'.

```
>>> ODD.keys()
dict_keys([1, 3, 5, 7, 9])
```

(c) Display the values in dictionary 'ODD'.

```
>>> ODD.values()
dict_values(['One', 'Three', 'Five', 'Seven', 'Nine'])
```

(d) Display the items from dictionary 'ODD'

```
>>> ODD.items()
dict_items([(1, 'One'), (3, 'Three'), (5, 'Five'), (7, 'Seven'), (9, 'Nine')])
```

(e) Find the length of the dictionary 'ODD'.

```
>>> len(ODD)
5
```

(f) Check if 7 is present or not in dictionary 'ODD'

```
>>> 7 in ODD
True
```

© NCERT not to be republished

(g) Check if 2 is present or not in dictionary 'ODD'

```
>>> 2 in ODD
False
```

(h) Retrieve the value corresponding to the key 9

```
>>> ODD.get(9)
'Nine'
```

(i) Delete the item from the dictionary, corresponding to the key 9. 'ODD'

```
>>> del ODD[9]
>>> ODD
{1: 'One', 3: 'Three', 5: 'Five', 7: 'Seven'}
```

Προγραμ 4–4 σ n number of write a program to enter names of employees and their salaries as input and store them in a dictionary. Here n is to input by the user.

```
#Program 4-4
#Program to create a dictionary which stores names of employees
#and their salary
num = int(input("Enter the number of employees whose data to be
stored: "))
count = 1
employee = dict()                    #create an empty dictionary
for count in range (n):
     name = input("Enter the name of the Employee: ")
     salary = int(input("Enter the salary: "))
     employee[name] = salary
print("\n\nEMPLOYEE_NAME\tSALARY")
for k in employee:
     print(k,'\t\t',employee[k])
```

Output:

```
Enter the number of employees to be stored: 5
Enter the name of the Employee: 'Tarun'
Enter the salary: 12000
Enter the name of the Employee: 'Amina'
Enter the salary: 34000
Enter the name of the Employee: 'Joseph'
Enter the salary: 24000
Enter the name of the Employee: 'Rahul'
Enter the salary: 30000
Enter the name of the Employee: 'Zoya'
Enter the salary: 25000
EMPLOYEE_NAME    SALARY
'Tarun'          12000
'Amina'          34000
```

© NCERT not to be republished

```
'Joseph'          24000
'Rahul'           30000
'Zoya'            25000
```

Program 4-5 Write a program to count the number of times a character appears in a given string.

```
#Program 4-5
#Count the number of times a character appears in a given string
st = input("Enter a string: ")
dic = {}                    #creates an empty dictionary
for ch in st:
     if ch in dic:          #if next character is already in dic
          dic[ch] += 1
     else:
          dic[ch] = 1       #if ch appears for the first time

for key in dic:
     print(key,':',dic[key])
```

Output:

```
Enter a string: HelloWorld
H : 1
e : 1
l : 3
o : 2
W : 1
r : 1
d : 1
```

Program 4-6 Write a program to convert a number entered by the user into its corresponding number in words. for example if the input is 876 then the output should be 'Eight Seven Six'.

```
# Program 4-6
num = input("Enter any number: ")     #number is stored as string
    #numberNames is a dictionary of digits and corresponding number
     #names
     numberNames = {0:'Zero',1:'One',2:'Two',3:'Three',4:'Four',\
                    5:'Five',6:'Six',7:'Seven',8:'Eight',9:'Nine'}

     result = ''
     for ch in num:
          key = int(ch)               #converts character to integer
          value = numberNames[key]
```

© NCERT not to be republished

```
            result = result + ' ' + value
    print("The number is:",num)
    print("The numberName is:",result)
```

Output:

```
Enter any number: 6512
The number is: 6512
The numberName is: Six Five One Two
```

SUMMARY

- Lists are mutable sequences in Python, i.e. we can change the elements of the list.
- Elements of a list are put in square brackets separated by comma.
- List indexing is same as that of list and starts at 0. Two way indexing allows traversing the list in the forward as well as in the backward direction.
- Operator + concatenates one list to the end of other list.
- Operator * repeats the content of a list by specified number of times.
- Membership operator `in` tells if an element is present in the list or not and `not in` does the opposite.
- Slicing is used to extract a part of the list.
- There are many list manipulation methods. Few are: len(), list(), append(), extend(), insert(), count(), find(), remove(), pop(), reverse(), sort(), sorted(), min(), max(), sum().
- Dictionary is a mapping (non scalar) data type. It is an unordered collection of key-value pair; key-value pair are put inside curly braces.
- Each key is separated from its value by a colon.
- Keys are unique and act as the index.
- Keys are of immutable type but values can be mutable.

© NCERT not to be republished

NOTES

EXERCISE

1. What will be the output of the following statements?

   a)
   ```
   list1 = [12,32,65,26,80,10]
   list1.sort()
   print(list1)
   ```

   b)
   ```
   list1 = [12,32,65,26,80,10]
   sorted(list1)
   print(list1)
   ```

   c)
   ```
   list1 = [1,2,3,4,5,6,7,8,9,10]
   list1[::-2]
   list1[:3] + list1[3:]
   ```

   d)
   ```
   list1 = [1,2,3,4,5]
   list1[len(list1)-1]
   ```

2. Consider the following list `myList`. What will be the elements of `myList` after each of the following operations?
   ```
   myList = [10,20,30,40]
   ```
   a) `myList.append([50,60])`
   b) `myList.extend([80,90])`

3. What will be the output of the following code segment?
   ```
   myList = [1,2,3,4,5,6,7,8,9,10]
   for i in range(0,len(myList)):
       if i%2 == 0:
           print(myList[i])
   ```

4. What will be the output of the following code segment?

   a)
   ```
   myList = [1,2,3,4,5,6,7,8,9,10]
   del myList[3:]
   print(myList)
   ```

   b)
   ```
   myList = [1,2,3,4,5,6,7,8,9,10]
   del myList[:5]
   print(myList)
   ```

   c)
   ```
   myList = [1,2,3,4,5,6,7,8,9,10]
   del myList[::2]
   print(myList)
   ```

5. Differentiate between append() and extend() methods of list.

© NCERT not to be republished

NOTES

6. Consider a list:

```
list1 = [6,7,8,9]
```

What is the difference between the following operations on list1:

a) `list1 * 2`

b) `list1 *= 2`

c) `list1 = list1 * 2`

7. The record of a student (Name, Roll No, Marks in five subjects and percentage of marks) is stored in the following list:

```
stRecord = ['Raman','A-36',[56,98,99,72,69],
            78.8]
```

Write Python statements to retrieve the following information from the list `stRecord`.

a) Percentage of the student

b) Marks in the fifth subject

c) Maximum marks of the student

d) Roll No. of the student

e) Change the name of the student from 'Raman' to 'Raghav'

8. Consider the following dictionary stateCapital:

```
stateCapital = {"Assam":"Guwahati",
"Bihar":"Patna","Maharashtra":"Mumbai",
"Rajasthan":"Jaipur"}
```

Find the output of the following statements:

a) `print(stateCapital.get("Bihar"))`

b) `print(stateCapital.keys())`

c) `print(stateCapital.values())`

d) `print(stateCapital.items())`

e) `print(len(stateCapital))`

f) `print("Maharashtra" in stateCapital)`

g) `print(stateCapital.get("Assam"))`

h)
```
del stateCapital["Assam"]
print(stateCapital)
```

PROGRAMMING PROBLEMS

1. Write a program to find the number of times an element occurs in the list.

© NCERT not to be republished

NOTES

2. Write a program to read a list of n integers (positive as well as negative). Create two new lists, one having all positive numbers and the other having all negative numbers from the given list. Print all three lists.
3. Write a program to find the largest and the second largest elements in a given list of elements.
4. Write a program to read a list of n integers and find their median.

   Note: The median value of a list of values is the middle one when they are arranged in order. If there are two middle values then take their average.

   Hint: Use an inbuilt function to sort the list.
5. Write a program to read a list of elements. Modify this list so that it does not contain any duplicate elements i.e. all elements occurring multiple times in the list should appear only once.
6. Write a program to create a list of elements. Input an element from the user that has to be inserted in the list. Also input the position at which it is to be inserted.
7. Write a program to read elements of a list and do the following.
   a) The program should ask for the position of the element to be deleted from the list and delete the element at the desired position in the list.
   b) The program should ask for the value of the element to be deleted from the list and delete this value from the list.
8. Write a Python program to find the highest 2 values in a dictionary.
9. Write a Python program to create a dictionary from a string 'w3resource' such that each individual character mates a key and its index value for fist occurrence males the corresponding value in dictionary.

   Expected output : {'3': 1, 's': 4, 'r': 2, 'u': 6, 'w': 0, 'c': 8,
   'e': 3, 'o': 5}
10. Write a program to input your friend's, names and their phone numbers and store them in the dictionary as the key-value pair. Perform the following operations on the dictionary:
    a) Display the Name and Phone number for all your friends.
    b) Add a new key-value pair in this dictionary and display the modified dictionary

© NCERT not to be republished

NOTES

c) Delete a particular friend from the dictionary

d) Modify the phone number of an existing friend

e) Check if a friend is present in the dictionary or not

f) Display the dictionary in sorted order of names

CASE STUDY BASED QUESTION

For the SMIS System given in Chapter 3, let us do the following:

1. Write a program to take in the roll number, name and percentage of marks for n students of Class X and do the following:
   - Accept details of the n students (n is the number of students).
   - Search details of a particular student on the basis of roll number and display result.
   - Display the result of all the students.
   - Find the topper amongst them.
   - Find the subject toppers amongst them.

   (**Hint:** Use Dictionary, where the key can be roll number and the value an immutable data type containing name and percentage.)

CASE STUDY

1. A bank is a financial institution which is involved in borrowing and lending of money. With advancement in technology, online banking, also known as internet banking allows customers of a bank to conduct a range of financial transactions through the bank's website anytime, anywhere. As part of initial investigation you are suggested to:
   - Collect a Bank's application form. After careful analysis of the form, identify the information required for opening a savings account. Also enquire about the rate of interest offered for a savings account.
   - The basic two operations performed on an account are Deposit and Withdrawal. Write a menu driven program that accepts either of the two choices of Deposit and Withdrawal, then accepts an amount, performs the transaction and accordingly displays the balance. Remember every bank has a requirement of minimum balance which needs to be taken care of during withdrawal operations.

© NCERT not to be republished

Notes

Enquire about the minimum balance required in your bank.

- Collect the interest rates for opening a fixed deposit in various slabs in a savings bank account. Remembers rate may be different for senior citizens.

Finally, write a menu driven program having the following options (use functions and appropriate data types):

- Open a savings bank account
- Deposit money
- Withdraw money
- Take details such as amount and period for a Fixed Deposit and display its maturity amount for a particular customer.

2. Participating in a quiz can be fun as it provides a competitive element. Some educational institutes use it as a tool to measure knowledge level, abilities and/or skills of their pupils either on a general level or in a specific field of study. Identify and analyse popular quiz shows and write a Python program to create a quiz that should also contain the following functionalities besides the one identified by you as a result of your analysis.
   - Create an administrative user ID and password to categorically add or modify delete a question.
   - Register the student before allowing her/him to play a quiz.
   - Allow selection of category based on subject area.
   - Display questions as per the chosen category.
   - Keep the score as the participant plays.
   - Display final score.
3. Our heritage monuments are our assets. They are a reflection of our rich and glorious past and an inspiration for our future. UNESCO has identified some of Indian heritage sites as World Heritage sites. Collect the following information about these sites:
   - What is the name of the site?
   - Where is it located?
     - District
     - State

© NCERT not to be republished

NOTES

- When was it built?
- Who built it?
- Why was it built?
- Website link (if any)

Write a Python program to:

- Create an administrative user ID and password to add, modify or delete an entered heritage site in the list of sites.
- Display the list of world heritage sites in India.
- Search and display information of a world heritage site entered by the user.
- Display the name(s) of world heritage site(s) on the basis of the state input by the user.

© NCERT not to be republished

11149CH05

Chapter 5

Understanding Data

"Data is not information, Information is not knowledge, Knowledge is not understanding, Understanding is not wisdom."

— *Gary Schubert*

In this chapter

- Introduction to Data
- Data Collection
- Data Storage
- Data Processing
- Statistical Techniques for Data Processing

5.1 Introduction to Data

Many a time, people take decisions based on certain data or information. For example, while choosing a college for getting admission, one looks at placement data of previous years of that college, educational qualification and experience of the faculty members, laboratory and hostel facilities, fees, etc. So we can say that identification of a college is based on various data and their analysis. Governments systematically collect and record data about the population through a process called census. Census data contains

© NCERT not to be republished

valuable information which are helpful is planning and formulating policies. Likewise, the coaching staff of a sports team analyses previous performances of opponent teams for making strategies. Banks maintain data about the customers, their account details and transactions. All these examples highlight the need of data in various fields. Data are indeed crucial for decision making.

In the previous examples, one cannot make decisions by looking at the data itself. In our example of choosing a college, suppose the placement cell of the college has maintained data of about 2000 students placed with different companies at different salary packages in the last 3 years. Looking at such data, one cannot make any remark about the placement of students of that college. The college processes and analyses this data and the results are given in the placement brochure of the college through summarisation as well as visuals for easy understanding. Hence, data need to be gathered, processed and analysed for making decisions.

A knowledge base is a store of information consisting of facts, assumptions and rules which an AI system can use for decision making.

In general, data is a collection of characters, numbers, and other symbols that represents values of some situations or variables. Data is plural and singular of the word data is "datum". Using computers, data are stored in electronic forms because data processing becomes faster and easier as compared to manual data processing done by people. The Information and Communication Technology (ICT) revolution led by computer, mobile and Internet has resulted in generation of large volume of data and at a very fast pace. The following list contains some examples of data that we often come across.

- Name, age, gender, contact details, etc., of a person
- Transactions data generated through banking, ticketing, shopping, etc. whether online or offline
- Images, graphics, animations, audio, video
- Documents and web pages
- Online posts, comments and messages
- Signals generated by sensors
- Satellite data including meteorological data, communication data, earth observation data, etc.

5.1.1 Importance of Data

Human beings rely on data for making decisions. Besides, large amount of data when processed with the help of a computer, show us the possibilities or hidden

© NCERT not to be republished

NOTES

traits which are otherwise not visible to humans. When one withdraws money from ATM, the bank needs to debit the withdrawn amount from the linked account. So the bank needs to maintain data and update it as and when required. The meteorological offices continuously keep on monitoring satellite data for any upcoming cyclone or heavy rain.

In a competitive business environment, it is important for business organisations to continuously monitor and analyse market behavior with respect to their products and take actions accordingly. Besides, companies identify customer demands as well as feedbacks, and make changes in their products or services accordingly.

The dynamic pricing concept used by airlines and railway is another example where they decide the price based on relationships between demand and supply. The cab booking Apps increase or decrease the price based on demand for cabs at a particular time. Certain restaurants offer discounted price (called happy hours), they decide when and how much discount to offer by analysing sales data at different time periods.

Besides business, following are some other scenarios where data are also stored and analysed for making decisions:

- The electronic voting machines are used for recording the votes cast. Subsequently, the voting data from all the machines are accumulated to declare election results in a short time as compared to manual counting of ballot papers.
- Scientists record data while doing experiments to calculate and compare results.
- Pharmaceutical companies record data while trying out a new medicine to see its effectiveness.
- Libraries maintain data about books in the library and the membership of the library.
- The search engines give us results after analysing large volume of data available on the websites across World Wide Web (www).
- Weather alerts are generated by analysing data received from various satellites.

5.1.2 Types of Data

As data come from different sources, they can be in different formats. For example, an image is a collection

© NCERT not to be republished

Activity 5.1

Observe Voter Identity cards of your family members and identify the data fields under which data are organised. Are they same for all?

of pixels; a video is made up of frames; a fee slip is made up of few numeric and non-numeric entries; and messages/chats are made up of texts, icons (emoticons) and images/videos. Two broad categories in which data can be classified on the basis of their format are:

(A) Structured Data

Data which is organised and can be recorded in a well defined format is called structured data. Structured data is usually stored in computer in a tabular (in rows and columns) format where each column represents different data for a particular parameter called attribute/characteristic/variable and each row represents data of an observation for different attributes. Table 5.1 shows structured data related to an inventory of kitchen items maintained by a shop.

Table 5.1 Structured data about kitchen items in a shop

ModelNo	ProductName	Unit Price	Discount(%)	Items_in_Inventory
ABC1	Water bottle	126	8	13
ABC2	Melamine Plates	320	5	45
ABC3	Dinner Set	4200	10	8
GH67	Jug	80	0	10
GH78	Table Spoon	120	5	14
GH81	Bucket	190	12	6
NK2	Kitchen Towel	25	0	32

Given this data, using a spreadsheet or other such software, the shop owner can find out how many total items are there by summing the column Items_in_Inventory of Table 5.1 The owner of the shop can also calculate the total value of all items in the inventory by multiplying each entry of column 3 (Unit Price) with corresponding entry of column 5 (Items_in_Inventory) and finding their sum.

Table 5.2 shows more examples of structured data recorded for different attributes.

Table 5.2 Attributes maintained for different activities

Entity/Activities	Data Fields/Parameters/Attributes
Books at a shop	BookTitle, Author, Price, YearofPublication
Depositing fees in a school	StudentName, Class, RollNo, FeesAmount, DepositDate
Amount withdrawal from ATM	AccHolderName, AccountNo, TypeofAcc, DateofWithdrawal, AmountWithdrawn, ATMid, TimeOfWithdrawal

© NCERT not to be republished

(B) Unstructured Data

A newspaper contains various types of news items which are also called data. But there is no fixed pattern that a newspaper follows in placing news articles. One day there might be three images of different sizes on a page along with five news items and one or more advertisements. While on another day there, might be one big image with three textual news items. So there is no particular format nor any fixed structure for printing news. Another example is the content of an email. There is no fixed structure about how many lines or paragraphs one has to write in an email or how many files are to be attached with an email. In summary, data which are not in the traditional row and column structure is called unstructured data.

Examples of unstructured data include web pages consisting of text as well as multimedia contents (image, graphics, audio/video). Other examples include text documents, business reports, books, audio/video files, social media messages. Although there are ways to process unstructured data, we are going to focus on handling structured data only in this book.

Unstructured data are sometimes described with the help of some other data called metadata. Metadata is basically data about data. For example, we describe different parts of an email as subject, recipient, main body, attachment, etc. These are the metadata for the email data. Likewise, we can have some metadata for an image file as image size (in KB or MB), image type (for example, JPEG, PNG), image resolution, etc.

Think and Reflect

When we click a photograph using our digital or mobile camera, does it have some metadata associated with it?

5.2 DATA COLLECTION

For processing data, we need to collect or gather data first. We can then store the data in a file or database for later use. Data collection here means identifying already available data or collecting from the appropriate sources. Suppose there are three different scenarios where sales data in a grocery store are available:

- Sales data are available with the shopkeeper in a diary or register. In this case we should enter the data in a digital format for example, in a spreadsheet.
- Data are already available in a digital format, say in a CSV (comma separated values) file.
- The shopkeeper has so far not recorded any data in either form but wants to get a software developed for

© NCERT not to be republished

Think and Reflect

Identify attributes needed for creating an Aadhaar Card.

maintaining sales data and accounts. The software may be developed using a programming language such as Python which can be used to store and retrieve data from a CSV file or a database management system like MySQL, which will be discussed further.

Data are continuously being generated at different sources. Our interactions with digital medium are continuously generating huge volumes of data. Hospitals are collecting data about patients for improving their services. Shopping malls are collecting data about the items being purchased by people. On analysing such data, suppose it appears that bedsheets and groceries are frequently bought together. Hence, the shop owner may decide to display bedsheets near the grocery section in the mall to increase the sales. Likewise, a political analyst may look at the data contained in the posts and messages at a social media platform and analyse to see public opinion before an election. Organisations like World Bank and International Monetary Fund (IMF) are collecting data related to various economic parameters from different countries for making economic forecasts.

5.3 DATA STORAGE

Once we gather data and process them to get results, we may not then simply discard the data. Rather, we would like to store them for future use as well. Data storage is the process of storing data on storage devices so that data can be retrieved later. Now a days large volume of data are being generated at a very high rate. As a result, data storage has become a challenging task. However, the decrease in the cost of digital storage devices has helped in simplifying this task. There are numerous digital storage devices available in the market like, Hard Disk Drive (HDD), Solid State Drive (SSD), CD/DVD, Tape Drive, Pen Drive, Memory Card, etc.

Think and Reflect

Is it necessary to store data in files before processing?

We store data like images, documents, audios/ videos, etc. as files in our computers. Likewise, school/ hospital data are stored in data files. We use computers to add, modify or delete data in these files or process these data files to get results. However, file processing has certain limitations, which can be overcome through Database Management System (DBMS).

© NCERT not to be republished

5.4 DATA PROCESSING

We are interested in understanding data as they hold valuable facts and information that can be useful in our decision making process. However, by looking at the vast or large amount of data, one cannot arrive at a conclusion. Rather, data need to be processed to get results and after analysing those results, we make conclusions or decisions.

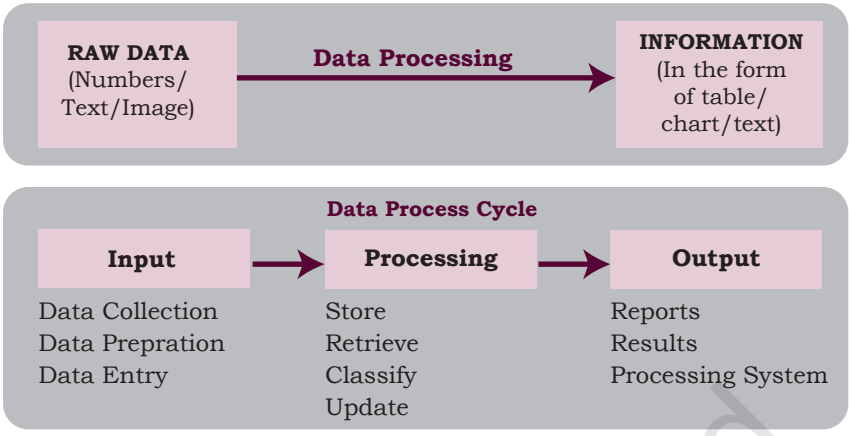

Figure 5.1: Steps in Data Processing

We find automated data processing in situations like online bill payment, registration of complaints, booking tickets, etc. Figure 5.1 illustrates basic steps used to process the data to get the output.

Figure 5.2 shows some tasks along with data, processing and generated output/information.

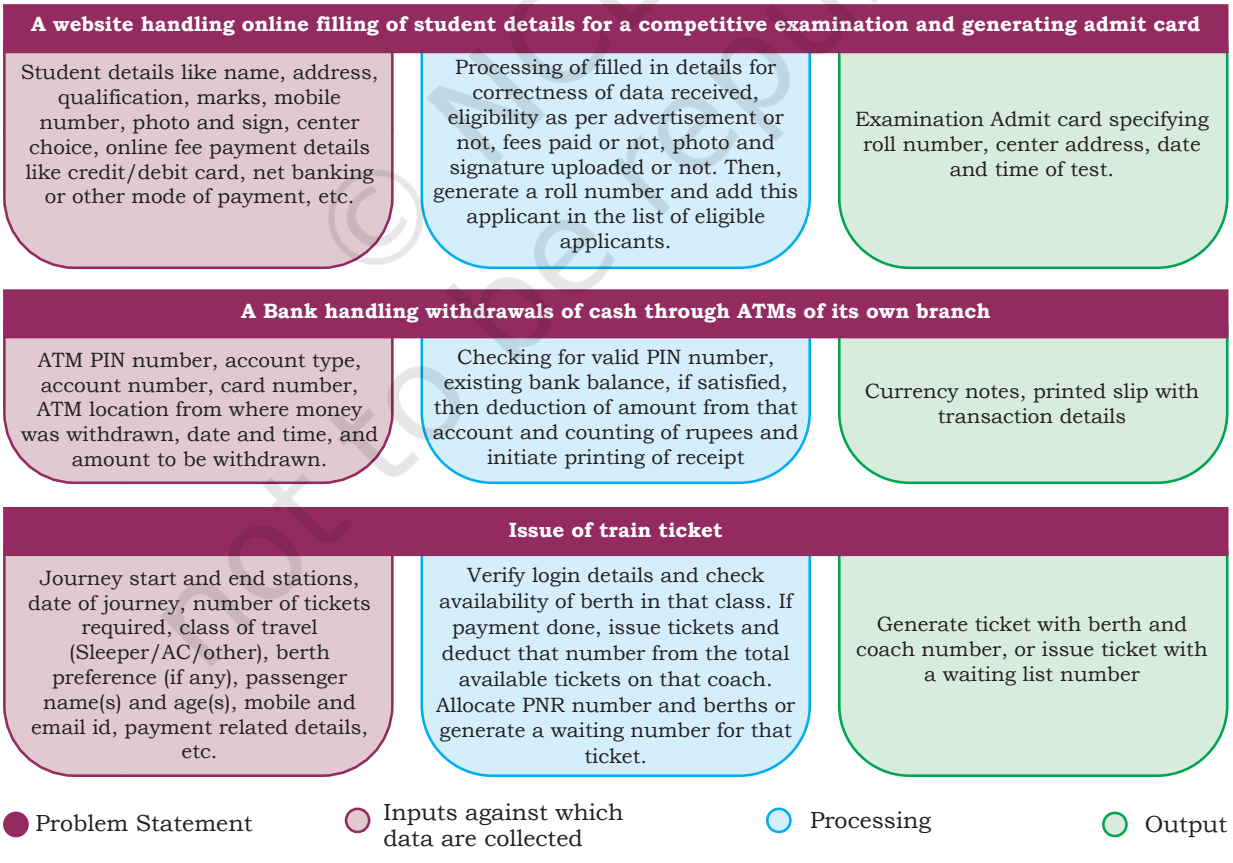

Figure 5.2: Data Based Problem Statements

© NCERT not to be republished

NOTES

5.5 STATISTICAL TECHNIQUES FOR DATA PROCESSING

Given a set of data values, we need to process them to get information. There are various techniques which help us to have preliminary understanding about the data. Summarisation methods are applied on tabular data for its easy comprehension. Commonly used statistical techniques for data summarisation are given below:

5.5.1 Measures of Central Tendency

A measure of central tendency is a single value that gives us some idea about the data. Three most common measures of central tendency are the *mean*, *median*, and *mode*. Instead of looking at each individual data values, we can calculate the mean, median and mode of the data to get an idea about average, middle value and frequency of occurrence of a particular value, respectively. Selection of a measure of central tendency depends on certain characteristics of data.

(A) Mean

Mean is simply the average of numeric values of an attribute. Mean is also called *average*. Suppose there are data on weight of 40 students in a class. Instead of looking at each of the data values, we can calculate the average to get an idea about the average weight of students in that class.

Definition: Given n values x_1, x_2, x_3,...x_n, mean is computed as $\frac{\sum_{i}^{n} x_i}{n}$.

Example 5.1

Assume that height (in cm) of students in a class are as follows [90,102,110,115,85,90,100,110,110]. Mean or average height of the class is

$$\frac{90+102+110+115+85+90+100+110+110}{9} = \frac{912}{9} = 101.33\,\text{cm}$$

Mean is not a suitable choice if there are outliers in the data. To calculate mean, the outliers or extreme values should be removed from the given data and then calculate mean of the remaining data.

Note: An outlier is an exceptionally large or small value, in comparison to other values of the data. Usually, outliers are considered as error since they can influence/affect the average or other statistical calculation based on the data.

© NCERT not to be republished

(B) Median

Median is also computed for a single attribute/variable at a time. When all the values are sorted in ascending or descending order, the middle value is called the *Median*. When there are odd number of values, then median is the value at the middle position. If the list has even number of values, then median is the average of the two middle values. Median represents the central value at which the given data is equally divided into two parts.

Example 5.2

Consider the previous data of height of students used in calculation of mean value. In order to compute the median, the first step is to sort data in ascending or descending order. We have sorted the height data in ascending order as [85,90,90,100,102,110,110,110, 115]. As there are total 9 values (odd number), the median is the value at position 5, that is 102 cm, whether counted from left to right or from right to left. Median represents the actual central value at which the given data is equally divided into two parts.

Think and Reflect

Out of Mean and Median, which one is more sensitive to outliers in data?

(C) Mode

Value that appears most number of times in the given data of an attribute/variable is called *Mode*. It is computed on the basis of frequency of occurrence of distinct values in the given data. A data set has no mode if each value occurs only once. There may be multiple modes in the data if more than one values have same highest frequency. Mode can be found for numeric as well as non-numeric data.

Example 5.3

In the list of height of students, mode is 110 as its frequency of occurrence in the list is 3, which is larger than the frequency of rest of the values.

5.5.2 Measures of Variability

The measures of variability refer to the spread or variation of the values around the mean. They are also called measures of dispersion that indicate the degree of diversity in a data set. They also indicate difference within the group. Two different data sets can have the same mean, median or mode but completely different levels of dispersion, or vice versa. Common measures of dispersion or variability are Range and Standard Deviation.

© NCERT not to be republished

NOTES

(A) Range

It is the difference between maximum and minimum values of the data (the largest value minus the smallest value). *Range* can be calculated only for numerical data. It is a measure of dispersion and tells about coverage/spread of data values. For example difference in salaries of employees, marks of a student, price of toys, etc. As range is calculated based on the two extreme values, any outlier in the data badly influences the result.

Let *M* be the largest or maximum value and *S* is the smallest or minimum value in the data, then Range is the difference between two extreme values i.e. *M – S or Maximum – Minimum.*

Example 5.4

In the above example, minimum hight value is 85 cm and maximum hight value is 115 cm. Hane range is 115-85 = 30 cm.

(B) Standard deviation

Standard deviation refers to differences within the group or set of data of a variable. Like Range, it also measures the spread of data. However, unlike Range which only uses two extreme values in the data, calculation of standard deviation considers all the given data. It is calculated as the positive square root of the average of squared difference of each value from the mean value of data. Smaller value of standard deviation means data are less spread while a larger value of standard deviation means data are more spread.

Given n values x_1, x_2, x_3,...x_n, and their mean $\bar{x}$, the standard deviation, represented as σ (greek letter sigma) is computed as

$$\Sigma = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{n}}$$

Example 5.5

Let us compute the standard deviation of the hight of nine students that we used while calculating Mean. The Mean ($\bar{x}$) was calculated to be 101.33 cm. Subtract each value from the mean and take square of that value. Dividing the sum of square values by total number of values and taking its square not gives the standard deviation in data. See Table 5.3 for details.

© NCERT not to be republished

Table 5.3 Standard deviation of attendance of 9 students

Height (x) in cm	$x - \bar{x}$	$(x - \bar{x})^2$
90	-11.33	128.37
102	0.67	0.36
110	8.67	75.17
115	13.67	186.87
85	-16.33	266.67
90	-11.33	128.37
100	-1.33	1.77
110	8.67	75.17
110	8.67	75.17
$n = 9$ $\bar{x} = 101.33$	$\sum(x-\bar{x}) = 0.03$	$\sum(x-\bar{x})^2 = 938.00$

$$= \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n}$$

$$= \frac{938}{9} = 104.22$$

$$\sum = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n}}$$

$$= \sqrt{104.22} = 10.2\,\text{cm}$$

Let us look at the following problems and select the suitable statistical technique to be applied (Mean/ Median/Mode/Range/Standard Deviation):

Problem Statement	Choose suitable statistical method
The management of a company wants to know about disparity in salaries of all employees.	
Teacher wants to know about the average performance of the whole class in a test.	
Compare height of residents of two cities	
Find the dominant value from a set of values	
Compare income of residents of two cities	
Find the popular color for car after surveying the car owners of a small city.	

It is important to understand statistical techniques so that one can decide which statistical technique to use to arrive at a decision. Different programming tools are available for efficient analysis of large volumes of data. These tools make use of statistical techniques for data analysis. One such programming tool is Python and it has libraries specially built for data processing and analysis. We will be covering some of them in the following chapters.

© NCERT not to be republished

NOTES

SUMMARY

- Data refer to unorganised facts that can be processed to generate meaningful result or information.
- Data can be structured or unstructured.
- Hard Disk, SSD, CD/DVD, Pen Drive, Memory Card, etc. are some of the commonly used storage devices.
- Data Processing cycle involves input and storage of data, its processing and generating output.
- Summarizing data using statistical techniques aids in revealing data characteristics.
- Mean, Median, Mode, Range, and Standard Deviation are some of the statistical techniques used for data summarisation.
- Mean is the average of given values.
- Median is the mid value when data are sorted in ascending/descending order.
- Mode is the data value that appears most number of times.
- Range is the difference between the maximum and minimum values.
- Standard deviation is the positive square root of the average of squared difference of each value from the mean.

© NCERT not to be republished

EXCERCISE

1. Identify data required to be maintained to perform the following services:
   a) Declare exam results and print e-certificates
   b) Register participants in an exhibition and issue biometric ID cards
   c) To search for an image by a search engine
   d) To book an OPD appointment with a hospital in a specific department
2. A school having 500 students wants to identify beneficiaries of the merit-cum means scholarship, achieving more than 75% for two consecutive years and having family income less than 5 lakh per annum.

NOTES

Briefly describe data processing steps to be taken by the to beneficial prepare the list of school.

3. A bank 'xyz' wants to know about its popularity among the residents of a city 'ABC' on the basis of number of bank accounts each family has and the average monthly account balance of each person. Briefly describe the steps to be taken for collecting data and what results can be checked through processing of the collected data.
4. Identify type of data being collected/generated in the following scenarios:
   a) Recording a video
   b) Marking attendance by teacher
   c) Writing tweets
   d) Filling an application form online
5. Consider the temperature (in Celsius) of 7 days of a week as 34, 34, 27, 28, 27, 34, 34. Identify the appropriate statistical technique to be used to calculate the following:
   a) Find the average temperature.
   b) Find the temperature Range of that week.
   c) Find the standard deviation temperature.
6. A school teacher wants to analyse results. Identify the appropriate statistical technique to be used along with its justification for the following cases:
   a) Teacher wants to compare performance in terms of division secured by students in Class XII A and Class XII B where each class strength is same.
   b) Teacher has conducted five unit tests for that class in months July to November and wants to compare the class performance in these five months.
7. Suppose annual day of your school is to be celebrated. The school has decided to felicitate those parents of the students studying in classes XI and XII, who are the alumni of the same school. In this context, answer the following questions:
   a) Which statistical technique should be used to find out the number of students whose both parents are alumni of this school?
   b) How varied are the age of parents of the students of that school?
8. For the annual day celebrations, the teacher is looking for an anchor in a class of 42 students. The teacher would make selection of an anchor on the basis of singing skill, writing skill, as well as monitoring skill.
   a) Which mode of data collection should be used?
   b) How would you represent the skill of students as data?

© NCERT not to be republished

NOTES

9. Differentiate between structured and unstructured data giving one example.
10. The principal of a school wants to do following analysis on the basis of food items procured and sold in the canteen:
    a) Compare the purchase and sale price of fruit juice and biscuits.
    b) Compare sales of fruit juice, biscuits and samosa.
    c) Variation in sale price of fruit juices of different companies for same quantity (in ml).

    Create an appropriate dataset for these items (fruit juice, biscuits, samosa) by listing their purchase price and sale price. Apply basic statistical techniques to make the comparisons.

© NCERT not to be republished

11149CH06

Chapter 6

Introduction to NumPy

"The goal is to turn data into information, and information into insight."

— *Carly Fiorina*

In this chapter

- Introduction
- Array
- NumPy Array
- Indexing and Slicing
- Operations on Arrays
- Concatenating Arrays
- Reshaping Arrays
- Splitting Arrays
- Statistical Operations on Arrays
- Loading Arrays from Files
- Saving NumPy Arrays in Files on Disk

6.1 Introduction

NumPy stands for 'Numerical Python'. It is a package for data analysis and scientific computing with Python. NumPy uses a multidimensional array object, and has functions and tools for working with these arrays. The powerful n-dimensional array in NumPy speeds-up data processing. NumPy can be easily interfaced with other Python packages and provides tools for integrating with other programming languages like C, C++ etc.

© NCERT not to be republished

Installing NumPy

NumPy can be installed by typing following command:

```
pip install NumPy
```

6.2 ARRAY

Contiguous memory allocation:
The memory space must be divided into the fined sized position and each position is allocated to a single data only.

Now Contiguous Memory Allocation:
Divide the data into several blocks and place in different parts of the memory according to the availability of memory space.

We have learnt about various data types like list, tuple, and dictionary. In this chapter we will discuss another datatype 'Array'. An array is a data type used to store multiple values using a single identifier (variable name). An array contains an ordered collection of data elements where each element is of the same type and can be referenced by its index (position).

The important characteristics of an array are:

- Each element of the array is of same data type, though the values stored in them may be different.
- The entire array is stored contiguously in memory. This makes operations on array fast.
- Each element of the array is identified or referred using the name of the Array along with the index of that element, which is unique for each element. The index of an element is an integral value associated with the element, based on the element's position in the array. For example consider an array with 5 numbers:

  [10, 9, 99, 71, 90]

Here, the 1st value in the array is 10 and has the index value [0] associated with it; the 2nd value in the array is 9 and has the index value [1] associated with it, and so on. The last value (in this case the 5th value) in this array has an index [4]. This is called zero based indexing. This is very similar to the indexing of lists in Python. The idea of arrays is so important that almost all programming languages support it in one form or another.

6.3 NUMPY ARRAY

NumPy arrays are used to store lists of numerical data, vectors and matrices. The NumPy library has a large set of routines (built-in functions) for creating, manipulating, and transforming NumPy arrays. Python language also has an array data structure, but it is not as versatile, efficient and useful as the NumPy array. The NumPy

© NCERT not to be republished

array is officially called ndarray but commonly known as array. In rest of the chapter, we will be referring to NumPy array whenever we use "array". following are few differences between list and Array.

6.3.1 Difference Between List and Array

List	Array
List can have elements of different data types for example, [1,3.4, 'hello', 'a@']	All elements of an array are of same data type for example, an array of floats may be: [1.2, 5.4, 2.7]
Elements of a list are not stored contiguously in memory.	Array elements are stored in contiguous memory locations. This makes operations on arrays faster than lists.
Lists do not support element wise operations, for example, addition, multiplication, etc. because elements may not be of same type.	Arrays support element wise operations. For example, if A1 is an array, it is possible to say A1/3 to divide each element of the array by 3.
Lists can contain objects of different datatype that Python must store the type information for every element along with its element value. Thus lists take more space in memory and are less efficient.	NumPy array takes up less space in memory as compared to a list because arrays do not require to store datatype of each element separately.
List is a part of core Python.	Array (ndarray) is a part of NumPy library.

6.3.2 Creation of NumPy Arrays from List

There are several ways to create arrays. To create an array and to use its methods, first we need to import the NumPy library.

```
#NumPy is loaded as np (we can assign any
#name), numpy must be written in lowercase
>>> import numpy as np
```

The NumPy's array() function converts a given list into an array. For example,

```
#Create an array called array1 from the
#given list.
>>> array1 = np.array([10,20,30])

#Display the contents of the array
>>> array1
array([10, 20, 30])
```

- Creating a 1-D Array

  An array with only single row of elements is called 1-D array. Let us try to create a 1-D array from a list which contains numbers as well as strings.

```
>>> array2 = np.array([5,-7.4,'a',7.2])
>>> array2
```

© NCERT not to be republished

```
array(['5', '-7.4', 'a', '7.2'],
dtype='<U32')
```

Observe that since there is a string value in the list, all integer and float values have been promoted to string, while converting the list to array.

Note: U32 means Unicode-32 data type.

- Creating a 2-D Array

  We can create a two dimensional (2-D) arrays by passing nested lists to the array() function.

A common mistake occurs while passing argument to array() if we forget to put square brackets. Make sure only a single argument containing list of values is passed.

```
#incorrect way
>>> a =
np.array(1,2,3,4)
#correct way
>>> a =
np.array([1,2,3,4])
```

Example 6.1

```
>>> array3 = np.array([[2.4,3],
                       [4.91,7],[0,-1]])
>>> array3
array([[ 2.4 ,  3.  ],
       [ 4.91,  7.  ],
       [ 0.  , -1.  ]])
```

Observe that the integers 3, 7, 0 and -1 have been promoted to floats.

6.3.3 Attributes of NumPy Array

Some important attributes of a NumPy ndarray object are:

i) `ndarray.ndim`: gives the number of dimensions of the array as an integer value. Arrays can be 1-D, 2-D or n-D. In this chapter, we shall focus on 1-D and 2-D arrays only. NumPy calls the dimensions as axes (plural of axis). Thus, a 2-D array has two axes. The row-axis is called axis-0 and the column-axis is called axis-1. The number of axes is also called the array's rank.

A list is called nested list when each element is a list itself.

Example 6.2

```
>>> array1.ndim
1
>>> array3.ndim
2
```

ii) `ndarray.shape`: It gives the sequence of integers indicating the size of the array for each dimension.

Example 6.3

```
# array1 is 1D-array, there is nothing
# after , in sequence
>>> array1.shape
(3,)
>>> array2.shape
(4,)
>>> array3.shape
(3, 2)
```

© NCERT not to be republished

NOTES

The output (3, 2) means array3 has 3 rows and 2 columns.

iii) `ndarray.size`: It gives the total number of elements of the array. This is equal to the product of the elements of shape.

Example 6.4

```
>>> array1.size
3
>>> array3.size
6
```

iv) `ndarray.dtype`: is the data type of the elements of the array. All the elements of an array are of same data type. Common data types are int32, int64, float32, float64, U32, etc.

Example 6.5

```
>>> array1.dtype
dtype('int32')
>>> array2.dtype
dtype('<U32>')
>>> array3.dtype
dtype('float64')
```

v) `ndarray.itemsize`: It specifies the size in bytes of each element of the array. Data type int32 and float32 means each element of the array occupies 32 bits in memory. 8 bits form a byte. Thus, an array of elements of type int32 has itemsize 32/8=4 bytes. Likewise, int64/float64 means each item has itemsize 64/8=8 bytes.

Example 6.6

```
>>> array1.itemsize
4           # memory allocated to integer
>>> array2.itemsize
128         # memory allocated to string
>>> array3.itemsize
8           #memory allocated to float type
```

6.3.4 Other Ways of Creating NumPy Arrays

1. We can specify data type (integer, float, etc.) while creating array using `dtype` as an argument to `array()`. This will convert the data automatically to the mentioned type. In the following example, nested list of integers are passed to the array function. Since data type has been declared as float, the integers are converted to floating point numbers.

© NCERT not to be republished

```
>>> array4 = np.array( [ [1,2], [3,4] ],
                       dtype=float)
>>> array4
array([[1., 2.],
       [3., 4.]])
```

2. We can create an array with all elements initialised to 0 using the function `zeros()`. By default, the data type of the array created by `zeros()` is float. The following code will create an array with 3 rows and 4 columns with each element set to 0.

```
>>> array5 = np.zeros((3,4))
>>> array5
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

3. We can create an array with all elements initialised to 1 using the function `ones()`. By default, the data type of the array created by `ones()` is float. The following code will create an array with 3 rows and 2 columns.

```
>>> array6 = np.ones((3,2))
>>> array6
array([[1., 1.],
       [1., 1.],
       [1., 1.]])
```

4. We can create an array with numbers in a given range and sequence using the `arange()` function. This function is analogous to the `range()` function of Python.

```
>>> array7 = np.arange(6)
# an array of 6 elements is created with
start value 5 and step size 1
>>> array7
array([0, 1, 2, 3, 4, 5])
# Creating an array with start value -2, end
# value 24 and step size 4
>>> array8 = np.arange( -2, 24, 4 )
>>> array8
array([-2,  2,  6, 10, 14, 18, 22])
```

6.4 INDEXING AND SLICING

NumPy arrays can be indexed, sliced and iterated over.

6.4.1 Indexing

We have learnt about indexing single-dimensional array in section 6.2. For 2-D arrays indexing for both dimensions starts from 0, and each element is referenced through two indexes i and j, where i represents the row number and j represents the column number.

Think and Reflect

When we may require to create an array initialised to zeros or ones?

© NCERT not to be republished

NOTES

Table 6.1 Marks of students in different subjects

Name	Maths	English	Science
Ramesh	78	67	56
Vedika	76	75	47
Harun	84	59	60
Prasad	67	72	54

Consider Table 6.1 showing marks obtained by students in three different subjects. Let us create an array called marks to store marks given in three subjects for four students given in this table. As there are 4 students (i.e. 4 rows) and 3 subjects (i.e. 3 columns), the array will be called `marks[4][3]`. This array can store 4*3 = 12 elements.

Here, `marks[i,j]` refers to the element at $(i+1)^{th}$ row and $(j+1)^{th}$ column because the index values start at 0. Thus `marks[3,1]` is the element in 4th row and second column which is 72 (marks of Prasad in English).

```
# accesses the element in the 1st row in
# the 3rd column
>>> marks[0,2]
56
>>> marks [0,4]
index Out of Bound "Index Error". Index 4
is out of bounds for axis with size 3
```

6.4.2 Slicing

Sometimes we need to extract part of an array. This is done through slicing. We can define which part of the array to be sliced by specifying the start and end index values using [start : end] along with the array name.

Example 6.7

```
>>> array8
array([-2,  2,  6, 10, 14, 18, 22])

# excludes the value at the end index
>>> array8[3:5]
array([10, 14])

# reverse the array
>>> array8[ : : -1]
array([22, 18, 14, 10,  6,  2, -2])
```

© NCERT not to be republished

NOTES

Now let us see how slicing is done for 2-D arrays. For this, let us create a 2-D array called array9 having 3 rows and 4 columns.

```
>>> array9 = np.array([[ -7, 0, 10, 20],
                       [ -5, 1, 40, 200],
                       [ -1, 1,  4, 30]])

# access all the elements in the 3rd column
>>> array9[0:3,2]
array([10, 40,  4])
```

Note that we are specifying rows in the range 0:3 because the end value of the range is excluded.

```
# access elements of 2nd and 3rd row from 1st
# and 2nd column
>>> array9[1:3,0:2]
array([[-5, 1],
       [-1, 1]])
```

If row indices are not specified, it means all the rows are to be considered. Likewise, if column indices are not specified, all the columns are to be considered. Thus, the statement to access all the elements in the 3rd column can also be written as:

```
>>>array9[:,2]
array([10, 40, 4])
```

6.5 OPERATIONS ON ARRAYS

Once arrays are declared, we con access it's element or perform certain operations the last section, we learnt about accessing elements. This section describes multiple operations that can be applied on arrays.

6.5.1 Arithmetic Operations

Arithmetic operations on NumPy arrays are fast and simple. When we perform a basic arithmetic operation like addition, subtraction, multiplication, division etc. on two arrays, the operation is done on each corresponding pair of elements. For instance, adding two arrays will result in the first element in the first array to be added to the first element in the second array, and so on. Consider the following element-wise operations on two arrays:

```
>>> array1 = np.array([[3,6],[4,2]])
>>> array2 = np.array([[10,20],[15,12]])
```

© NCERT not to be republished

NOTES

```
#Element-wise addition of two matrices.
>>> array1 + array2
array([[13, 26],
       [19, 14]])

#Subtraction
>>> array1 - array2
array([[ -7, -14],
       [-11, -10]])

#Multiplication
>>> array1 * array2
array([[ 30, 120],
       [ 60,  24]])

#Matrix Multiplication
>>> array1 @ array2
array([[120, 132],
       [ 70, 104]])

#Exponentiation
>>> array1 ** 3
array([[ 27, 216],
       [ 64,   8]], dtype=int32)

#Division
>>> array2 / array1
array([[3.33333333, 3.33333333],
       [3.75      , 6.        ]])

#Element wise Remainder of Division
#(Modulo)
>>> array2 % array1
array([[1, 2],
       [3, 0]], dtype=int32)
```

It is important to note that for element-wise operations, size of both arrays must be same. That is, array1.shape must be equal to array2.shape.

6.5.2 Transpose

Transposing an array turns its rows into columns and columns into rows just like matrices in mathematics.

```
#Transpose
>>> array3 = np.array([[10,-7,0, 20],
              [-5,1,200,40],[30,1,-1,4]])
>>> array3
array([[ 10,  -7,   0,  20],
       [ -5,   1, 200,  40],
       [ 30,   1,  -1,   4]])
```

© NCERT not to be republished

NOTES

```
# the original array does not change
>>> array3.transpose()
array([[ 10,  -5,  30],
       [ -7,   1,   1],
       [  0, 200,  -1],
       [ 20,  40,   4]])
```

6.5.3 Sorting

Sorting is to arrange the elements of an array in hierarchical order either ascending or descending. By default, numpy does sorting in ascending order.

```
>>> array4 = np.array([1,0,2,-3,6,8,4,7])
>>> array4.sort()
>>> array4
array([-3,  0,  1,  2,  4,  6,  7,  8])
```

In 2-D array, sorting can be done along either of the axes i.e., row-wise or column-wise. By default, sorting is done row-wise (i.e., on axis = 1). It means to arrange elements in each row in ascending order. When axis=0, sorting is done column-wise, which means each column is sorted in ascending order.

```
>>> array4 = np.array([[10,-7,0, 20],
                       [-5,1,200,40],[30,1,-1,4]])
>>> array4
array([[ 10,  -7,   0,  20],
       [ -5,   1, 200,  40],
       [ 30,   1,  -1,   4]])

#default is row-wise sorting
>>> array4.sort()
>>> array4
array([[ -7,   0,  10,  20],
       [ -5,   1,  40, 200],
       [ -1,   1,   4,  30]])
>>> array5 = np.array([[10,-7,0, 20],
                       [-5,1,200,40],[30,1,-1,4]])

#axis =0 means column-wise sorting
>>> array5.sort(axis=0)
>>> array5
array([[ -5,  -7,  -1,   4],
       [ 10,   1,   0,  20],
       [ 30,   1, 200,  40]])
```

6.6 CONCATENATING ARRAYS

Concatenation means joining two or more arrays. Concatenating 1-D arrays means appending the sequences one after another. NumPy.concatenate()

© NCERT not to be republished

NOTES

function can be used to concatenate two or more 2-D arrays either row-wise or column-wise. All the dimensions of the arrays to be concatenated must match exactly except for the dimension or axis along which they need to be joined. Any mismatch in the dimensions results in an error. By default, the concatenation of the arrays happens along axis=0.

Example 6.8

```
>>> array1 = np.array([[10, 20], [-30,40]])
>>> array2 = np.zeros((2, 3), dtype=array1.
             dtype)

>>> array1
array([[ 10,  20],
       [-30,  40]])

>>> array2
array([[0, 0, 0],
       [0, 0, 0]])

>>> array1.shape
(2, 2)
>>> array2.shape
(2, 3)

>>> np.concatenate((array1,array2), axis=1)
array([[ 10,  20,   0,   0,   0],
       [-30,  40,   0,   0,   0]])

>>> np.concatenate((array1,array2), axis=0)
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    np.concatenate((array1,array2))
ValueError: all the input array dimensions
except for the concatenation axis must
match exactly
```

6.7 RESHAPING ARRAYS

We can modify the shape of an array using the `reshape()` function. Reshaping an array cannot be used to change the total number of elements in the array. Attempting to change the number of elements in the array using `reshape()` results in an error.

Example 6.9

```
>>> array3 = np.arange(10,22)
>>> array3
array([10, 11, 12, 13, 14, 15, 16, 17, 18,
19, 20, 21])
```

© NCERT not to be republished

NOTES

```
>>> array3.reshape(3,4)
array([[10, 11, 12, 13],
       [14, 15, 16, 17],
       [18, 19, 20, 21]])

>>> array3.reshape(2,6)
array([[10, 11, 12, 13, 14, 15],
       [16, 17, 18, 19, 20, 21]])
```

6.8 SPLITTING ARRAYS

We can split an array into two or more subarrays. `numpy.split()` splits an array along the specified axis. We can either specify sequence of index values where an array is to be split; or we can specify an integer N, that indicates the number of equal parts in which the array is to be split, as parameter(s) to the `NumPy.split()` function. By default, `NumPy.split()` splits along axis = 0. Consider the array given below:

```
>>> array4
array([[ 10,  -7,   0,  20],
       [ -5,   1, 200,  40],
       [ 30,   1,  -1,   4],
       [  1,   2,   0,   4],
       [  0,   1,   0,   2]])

# [1,3] indicate the row indices on which
# to split the array
>>> first, second, third = numpy split(array4,
    [1, 3])

# array4 is split on the first row and
# stored on the sub-array first
>>> first
array([[10, -7,  0, 20]])

# array4 is split after the first row and
# upto the third row and stored on the
# sub-array second
>>> second
array([[ -5,   1, 200,  40],
       [ 30,   1,  -1,   4]])

# the remaining rows of array4 are stored
# on the sub-array third
>>> third
array([[1, 2, 0, 4],
       [0, 1, 0, 2]])
```

© NCERT not to be republished

NOTES

```
#[1, 2], axis=1 give the columns indices
#along which to split
>>> firstc, secondc, thirdc =numpy split(array4,
[1, 2], axis=1)
>>> firstc
array([[10],
       [-5],
       [30],
       [ 1],
       [ 0]])

>>> secondc
array([[-7],
       [ 1],
       [ 1],
       [ 2],
       [ 1]])

>>> thirdc
array([[  0,  20],
       [200,  40],
       [ -1,   4],
       [  0,   4],
       [  0,   2]])

# 2nd parameter 2 implies array is to be
# split in 2 equal parts axis=1 along the
# column axis
>>> firsthalf, secondhalf =np.split(array4,2,
axis=1)
>>> firsthalf
array([[10, -7],
       [-5,  1],
       [30,  1],
       [ 1,  2],
       [ 0,  1]])

>>> secondhalf
array([[  0,  20],
       [200,  40],
       [ -1,   4],
       [  0,   4],
       [  0,   2]])
```

6.9 STATISTICAL OPERATIONS ON ARRAYS

NumPy provides functions to perform many useful statistical operations on arrays. In this section, we will apply the basic statistical techniques called descriptive statistics that we have learnt in chapter 5.

© NCERT not to be republished

NOTES

Let us consider two arrays:

```
>>> arrayA = np.array([1,0,2,-3,6,8,4,7])
>>> arrayB = np.array([[3,6],[4,2]])
```

1. The `max()` function finds the maximum element from an array.

```
# max element form the whole 1-D array
>>> arrayA.max()
8
# max element form the whole 2-D array
>>> arrayB.max()
6
# if axis=1, it gives column wise maximum
>>> arrayB.max(axis=1)
array([6, 4])
# if axis=0, it gives row wise maximum
>>> arrayB.max(axis=0)
array([4, 6])
```

2. The `min()` function finds the minimum element from an array.

```
>>> arrayA.min()
-3
>>> arrayB.min()
2
>>> arrayB.min(axis=0)
array([3, 2])
```

3. The `sum()` function finds the sum of all elements of an array.

```
>>> arrayA.sum()
25
>>> arrayB.sum()
15
#axis is used to specify the dimension
#on which sum is to be made. Here axis = 1
#means the sum of elements on the first row
>>> arrayB.sum(axis=1)
array([9, 6])
```

4. The `mean()` function finds the average of elements of the array.

```
>>> arrayA.mean()
3.125
>>> arrayB.mean()
3.75
>>> arrayB.mean(axis=0)
array([3.5, 4. ])

>>> arrayB.mean(axis=1)
array([4.5, 3. ])
```

5. The `std()` function is used to find standard deviation of an array of elements.

```
>>> arrayA.std()
3.550968177835448
```

© NCERT not to be republished

NOTES

```
>>> arrayB.std()
1.479019945774904

>>> arrayB.std(axis=0)
array([0.5, 2. ])

>>> arrayB.std(axis=1)
array([1.5, 1. ])
```

6.10 LOADING ARRAYS FROM FILES

Sometimes, we may have data in files and we may need to load that data in an array for processing. `numpy.loadtxt()` and `numpy.genfromtxt()`are the two functions that can be used to load data from text files. The most commonly used file type to handle large amount of data is called CSV (Comma Separated Values).

Each row in the text file must have the same number of values in order to load data from a text file into a numpy array. Let us say we have the following data in a text file named `data.txt` stored in the folder `C:/NCERT`.

RollNo	Marks1	Marks2	Marks3
1,	36,	18,	57
2,	22,	23,	45
3,	43,	51,	37
4,	41,	40,	60
5,	13,	18,	37

We can load the data from the `data.txt` file into an array say, `studentdata` in the following manner:

6.10.1 Using NumPy.loadtxt()

```
>>> studentdata = np.loadtxt('C:/NCERT/
    data.txt', skiprows=1, delimiter=',',
    dtype = int)

>>> studentdata
array([[ 1, 36, 18, 57],
       [ 2, 22, 23, 45],
       [ 3, 43, 51, 37],
       [ 4, 41, 40, 60],
       [ 5, 13, 18, 27]])
```

In the above statement, first we specify the name and path of the text file containing the data. Let us understand some of the parameters that we pass in the `np.loadtext()` function:

© NCERT not to be republished

- The parameter skiprows=1 indicates that the first row is the header row and therefore we need to skip it as we do not want to load it in the array.
- The `delimiter` specifies whether the values are separated by comma, semicolon, tab or space (the four are together called whitespace), or any other character. The default value for delimiter is space.
- We can also specify the data type of the array to be created by specifying through the `dtype` argument. By default, `dtype` is float.

We can load each row or column of the data file into different numpy arrays using the unpack parameter. By default, `unpack=False` means we can extract each row of data as separate arrays. When `unpack=True`, the returned array is transposed means we can extract the columns as separate arrays.

```
# To import data into multiple NumPy arrays
# row wise. Values related to student1 in
# array stud1, student2 in array stud2 etc.
>>> stud1, stud2, stud3, stud4, stud5 =
np.loadtxt('C:/NCERT/data.txt',skiprows=1,
delimiter=',', dtype = int)

>>> stud1
array([ 1, 36, 18, 57])
>>> stud2
array([ 2, 22, 23, 45])  # and so on

# Import data into multiple arrays column
# wise. Data in column RollNo will be put
# in array rollno, data in column Marks1
# will be put in array mks1 and so on.
>>> rollno, mks1, mks2, mks3 =
np.loadtxt('C:/NCERT/data.txt',
skiprows=1, delimiter=',', unpack=True,
dtype = int)
>>> rollno
array([1, 2, 3, 4, 5])

>>> mks1
array([36, 22, 43, 41, 13])

>>> mks2
array([18, 23, 51, 40, 18])

>>> mks3
array([57, 45, 37, 60, 27])
```

.CSV files or comma separated values files are a type of text files that have values separated by commas. A CSV file stores tabular data in a text file. CSV files can be loaded in NumPy arrays and their data can be analyzed using these functions.

© NCERT not to be republished

6.10.2 Using NumPy.genfromtxt()

`genfromtxt()` is another function in NumPy to load data from files. As compared to `loadtxt()`, `genfromtxt()` can also handle missing values in the data file. Let us look at the following file `dataMissing.txt` with some missing values and some non-numeric data:

RollNo	Marks1	Marks2	Marks3
1,	36,	18,	57
2,	ab,	23,	45
3,	43,	51,	
4,	41,	40,	60
5,	13,	18,	27

```
>>> dataarray = np.genfromtxt('C:/NCERT/
    dataMissing.txt',skip_header=1,
    delimiter = ',')

>>> dataarray
array([[ 1., 36., 18., 57.],
       [ 2., nan, 23., 45.],
       [ 3., 43., 51., nan],
       [ 4., 41., 40., 60.],
       [ 5., 13., 18., 27.]])
```

The `genfromtxt()` function converts missing values and character strings in numeric columns to `nan`. But if we specify dtype as `int`, it converts the missing or other non numeric values to -1. We can also convert these missing values and character strings in the data files to some specific value using the parameter `filling_values`.

Example 6.10 Let us set the value of the missing or non numeric data to -999:

```
>>> dataarray = np.genfromtxt('C:/NCERT/
    dataMissing.txt',skip_header=1,
    delimiter=',', filling_values=-999,
    dtype = int)

>>> dataarray
array([[   1,   36,   18,   57],
       [   2, -999,   23,   45],
       [   3,   43,   51, -999],
       [   4,   41,   40,   60],
       [   5,   13,   18,   27]])
```

Activity 6.1

Can you write the command to load the data.txt including the header row as well?

Activity 6.2

Can you create a datafile and import data into multiple NumPy arrays column wise? (*Hint:* use unpack parameter)

© NCERT not to be republished

NOTES

6.11 SAVING NUMPY ARRAYS IN FILES ON DISK

The `savetxt()` function is used to save a NumPy array to a text file.

Example 6.11

```
>>> np.savetxt('C:/NCERT/testout.txt',
studentdata, delimiter=',', fmt='%i')
```

Note: We have used parameter `fmt` to specify the format in which data are to be saved. The default is float.

SUMMARY

- Array is a data type that holds objects of same datatype (numeric, textual, etc.). The elements of an array are stored contiguously in memory. Each element of an array has an index or position value.
- NumPy is a Python library for scientific computing which stores data in a powerful n-dimensional ndarray object for faster calculations.
- Each element of an array is referenced by the array name along with the index of that element.
- `numpy.array()` is a function that returns an object of type numpy.ndarray.
- All arithmetic operations can be performed on arrays when shape of the two arrays is same.
- NumPy arrays are not expandable or extendable. Once a numpy array is defined, the space it occupies in memory is fixed and cannot be changed.
- `numpy.split()` slices apart an array into multiple sub-arrays along an axis.
- `numpy.concatenate()` function can be used to concatenate arrays.
- `numpy.loadtxt()` and `numpy.genfromtxt()` are functions used to load data from files. The savetxt() function is used to save a NumPy array to a text file.

© NCERT not to be republished

NOTES

EXERCISE

1. What is NumPy ? How to install it?
2. What is an array and how is it different from a list? What is the name of the built-in array class in NumPy ?
3. What do you understand by rank of an ndarray?
4. Create the following NumPy arrays:
   a) A 1-D array called `zeros` having 10 elements and all the elements are set to zero.
   b) A 1-D array called `vowels` having the elements 'a', 'e', 'i', 'o' and 'u'.
   c) A 2-D array called ones having 2 rows and 5 columns and all the elements are set to 1 and dtype as int.
   d) Use nested Python lists to create a 2-D array called `myarray1` having 3 rows and 3 columns and store the following data:

      2.7, -2, -19
      0, 3.4, 99.9
      10.6, 0, 13

   e) A 2-D array called `myarray2` using `arange()` having 3 rows and 5 columns with start value = 4, step size 4 and dtype as float.
5. Using the arrays created in Question 4 above, write NumPy commands for the following:
   a) Find the dimensions, shape, size, data type of the items and itemsize of arrays `zeros, vowels, ones, myarray1 and myarray2.`
   b) Reshape the array ones to have all the 10 elements in a single row.
   c) Display the 2nd and 3rd element of the array `vowels.`
   d) Display all elements in the 2nd and 3rd row of the array `myarray1.`
   e) Display the elements in the 1st and 2nd column of the array `myarray1.`
   f) Display the elements in the 1st column of the 2nd and 3rd row of the array `myarray1.`
   g) Reverse the array of `vowels.`
6. Using the arrays created in Question 4 above, write NumPy commands for the following:

© NCERT not to be republished

NOTES

a) Divide all elements of array ones by 3.
b) Add the arrays `myarray1` and `myarray2`.
c) Subtract `myarray1` from `myarray2` and store the result in a new array.
d) Multiply `myarray1` and `myarray2` elementwise.
e) Do the matrix multiplication of `myarray1` and `myarray2` and store the result in a new array `myarray3`.
f) Divide `myarray1` by `myarray2`.
g) Find the cube of all elements of `myarray1` and divide the resulting array by 2.
h) Find the square root of all elements of `myarray2` and divide the resulting array by 2. The result should be rounded to two places of decimals.

7. Using the arrays created in Question 4 above, write NumPy commands for the following:
   a) Find the transpose of `ones` and `myarray2`.
   b) Sort the array `vowels` in reverse.
   c) Sort the array `myarray1` such that it brings the lowest value of the column in the first row and so on.
8. Using the arrays created in Question 4 above, write NumPy commands for the following:
   a) Use `NumPy. split()` to split the array `myarray2` into 5 arrays columnwise. Store your resulting arrays in `myarray2A, myarray2B, myarray2C, myarray2D` and `myarray2E`. Print the arrays `myarray2A, myarray2B, myarray2C, myarray2D` and `myarray2E`.
   b) Split the array `zeros` at array index 2, 5, 7, 8 and store the resulting arrays in `zerosA, zerosB, zerosC` and `zerosD` and print them.
   c) Concatenate the arrays `myarray2A, myarray2B` and `myarray2C` into an array having 3 rows and 3 columns.
9. Create a 2-D array called `myarray4` using `arange()` having 14 rows and 3 columns with start value = -1, step size 0.25 having. Split this array row wise into 3 equal parts and print the result.
10. Using the `myarray4` created in the above questions, write commands for the following:
    a) Find the sum of all elements.
    b) Find the sum of all elements row wise.

© NCERT not to be republished

NOTES

c) Find the sum of all elements column wise.
d) Find the max of all elements.
e) Find the min of all elements in each row.
f) Find the mean of all elements in each row.
g) Find the standard deviation column wise.

CASE STUDY (SOLVED)

We have already learnt that a data set (or dataset) is a collection of data. Usually a data set corresponds to the contents of a database table, or a statistical data matrix, where every column of the table represents a particular variable, and each row corresponds to a member or an item etc. A data set lists values for each of the variables, such as height and weight of a student, for each row (item) of the data set. Open data refers to information released in a publicly accessible repository.

The Iris flower data set is an example of an open data. It is also called Fisher's Iris data set as this data set was introduced by the British statistician and biologist Ronald Fisher in 1936. The Iris data set consists of 50 samples from each of the three species of the flower Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured for each sample: the length and the width of the sepals and petals, in centimeters. Based on the combination of these four features, Fisher developed a model to distinguish one species from each other. The full data set is freely available on UCI Machine Learning Repository at https://archive.ics.uci.edu/ml/datasets/iris.

We shall use the following smaller section of this data set having 30 rows (10 rows for each of the three species). We shall include a column for species number that has a value 1 for Iris setosa, 2 for Iris virginica and 3 for Iris versicolor.

Sepal Length	Sepal Width	Petal Length	Petal Width	Iris	Species No
5.1	3.5	1.4	0.2	Iris-setosa	1
4.9	3	1.4	0.2	Iris-setosa	1
4.7	3.2	1.3	0.2	Iris-setosa	1
4.6	3.1	1.5	0.2	Iris-setosa	1
5	3.6	1.4	0.2	Iris-setosa	1
5.4	3.9	1.7	0.4	Iris-setosa	1
4.6	3.4	1.4	0.3	Iris-setosa	1
5	3.4	1.5	0.2	Iris-setosa	1
4.4	2.9	1.4	0.2	Iris-setosa	1

© NCERT not to be republished

NOTES

4.9	3.1	1.5	0.1	Iris-setosa	1
5.5	2.6	4.4	1.2	Iris-versicolor	2
6.1	3	4.6	1.4	Iris-versicolor	2
5.8	2.6	4	1.2	Iris-versicolor	2
5	2.3	3.3	1	Iris-versicolor	2
5.6	2.7	4.2	1.3	Iris-versicolor	2
5.7	3	4.2	1.2	Iris-versicolor	2
5.7	2.9	4.2	1.3	Iris-versicolor	2
6.2	2.9	4.3	1.3	Iris-versicolor	2
5.1	2.5	3	1.1	Iris-versicolor	2
5.7	2.8	4.1	1.3	Iris-versicolor	2
6.9	3.1	5.4	2.1	Iris-virginica	3
6.7	3.1	5.6	2.4	Iris-virginica	3
6.9	3.1	5.1	2.3	Iris-virginica	3
5.8	2.7	5.1	1.9	Iris-virginica	3
6.8	3.2	5.9	2.3	Iris-virginica	3
6.7	3.3	5.7	2.5	Iris-virginica	3
6.7	3	5.2	2.3	Iris-virginica	3
6.3	2.5	5	1.9	Iris-virginica	3
6.5	3	5.2	2	Iris-virginica	3
6.2	3.4	5.4	2.3	Iris-virginica	3

You may type this using any text editor (`Notepad, gEdit or any other`) in the way as shown below and store the file with a name called `Iris.txt`. (In case you wish to work with the entire dataset you could download a .csv file for the same from the Internet and `save it as Iris.txt`). The headers are:

sepal length, sepal width, petal length, petal width, iris, Species No

5.1, 3.5, 1.4, 0.2, Iris-setosa, 1

4.9, 3, 1.4, 0.2, Iris-setosa, 1

4.7, 3.2, 1.3, 0.2, Iris-setosa, 1

4.6, 3.1, 1.5, 0.2, Iris-setosa, 1

5, 3.6, 1.4, 0.2, Iris-setosa, 1

5.4, 3.9, 1.7, 0.4, Iris-setosa, 1

4.6, 3.4, 1.4, 0.3, Iris-setosa, 1

5, 3.4, 1.5, 0.2, Iris-setosa, 1

4.4, 2.9, 1.4, 0.2, Iris-setosa, 1

4.9, 3.1, 1.5, 0.1, Iris-setosa, 1

© NCERT not to be republished

NOTES

5.5, 2.6, 4.4, 1.2, Iris-versicolor, 2
6.1, 3, 4.6, 1.4, Iris-versicolor, 2
5.8, 2.6, 4, 1.2, Iris-versicolor, 2
5, 2.3, 3.3, 1, Iris-versicolor, 2
5.6, 2.7, 4.2, 1.3, Iris-versicolor, 2
5.7, 3, 4.2, 1.2, Iris-versicolor, 2
5.7, 2.9, 4.2, 1.3, Iris-versicolor, 2
6.2, 2.9, 4.3, 1.3, Iris-versicolor, 2
5.1, 2.5, 3, 1.1, Iris-versicolor, 2
5.7, 2.8, 4.1, 1.3, Iris-versicolor, 2
6.9, 3.1, 5.4, 2.1, Iris-virginica, 3
6.7, 3.1, 5.6, 2.4, Iris-virginica, 3
6.9, 3.1, 5.1, 2.3, Iris-virginica, 3
5.8, 2.7, 5.1, 1.9, Iris-virginica, 3
6.8, 3.2, 5.9, 2.3, Iris-virginica, 3
6.7, 3.3, 5.7, 2.5, Iris-virginica, 3
6.7, 3, 5.2, 2.3, Iris-virginica, 3
6.3, 2.5, 5, 1.9, Iris-virginica, 3
6.5, 3, 5.2, 2, Iris-virginica, 3
6.2, 3.4, 5.4, 2.3, Iris-virginica, 3

1. Load the data in the file `Iris.txt` in a 2-D array called `iris`.
2. Drop column whose index = 4 from the array `iris`.
3. Display the shape, dimensions and size of `iris`.
4. Split `iris` into three 2-D arrays, each array for a different species. Call them `iris1, iris2, iris3`.
5. Print the three arrays `iris1, iris2, iris3`
6. Create a 1-D array `header` having elements "sepal length", "sepal width", "petal length", "petal width", "Species No" in that order.
7. Display the array `header`.
8. Find the max, min, mean and standard deviation for the columns of the iris and store the results in the arrays `iris_max, iris_min, iris_avg, iris_std, iris_var` respectively. The results must be rounded to not more than two decimal places.

© NCERT not to be republished

NOTES

9. Similarly find the max, min, mean and standard deviation for the columns of the `iris1`, `iris2` and `iris3` and store the results in the arrays with appropriate names.

10. Check the minimum value for sepal length, sepal width, petal length and petal width of the three species in comparison to the minimum value of sepal length, sepal width, petal length and petal width for the data set as a whole and fill the table below with True if the species value is greater than the dataset value and False otherwise.

	Iris setosa	Iris virginica	Iris versicolor
sepal length			
sepal width			
petal length			
petal width			

11. Compare Iris setosa's average sepal width to that of Iris virginica.

12. Compare Iris setosa's average petal length to that of Iris virginica.

13. Compare Iris setosa's average petal width to that of Iris virginica.

14. Save the array `iris_avg` in a comma separated file named `IrisMeanValues.txt` on the hard disk.

15. Save the arrays `iris_max`, `iris_avg`, `iris_min` in a comma separated file named `IrisStat.txt` on the hard disk.

SOLUTIONS TO CASE STUDY BASED EXERCISES

```
>>> import numpy as np

# Solution to Q1
>>> iris = np.genfromtxt('C:/NCERT/Iris.txt',skip_
    header=1, delimiter=',', dtype = float)

# Solution to Q2
>>> iris = iris[0:30,[0,1,2,3,5]] # drop column 4

# Solution to Q3
>>> iris.shape
(30, 5)
>>> iris.ndim
```

© NCERT not to be republished

NOTES

```
2
>>> iris.size
150

# Solution to Q4
# Split into three arrays, each array for a different
# species
>>> iris1, iris2, iris3 = np.split(iris, [10,20],
    axis=0)

# Solution to Q5
# Print the three arrays
>>> iris1
array([[5.1, 3.5, 1.4, 0.2, 1. ],
       [4.9, 3. , 1.4, 0.2, 1. ],
       [4.7, 3.2, 1.3, 0.2, 1. ],
       [4.6, 3.1, 1.5, 0.2, 1. ],
       [5. , 3.6, 1.4, 0.2, 1. ],
       [5.4, 3.9, 1.7, 0.4, 1. ],
       [4.6, 3.4, 1.4, 0.3, 1. ],
       [5. , 3.4, 1.5, 0.2, 1. ],
       [4.4, 2.9, 1.4, 0.2, 1. ],
       [4.9, 3.1, 1.5, 0.1, 1. ]])

>>> iris2
array([[5.5, 2.6, 4.4, 1.2, 2. ],
       [6.1, 3. , 4.6, 1.4, 2. ],
       [5.8, 2.6, 4. , 1.2, 2. ],
       [5. , 2.3, 3.3, 1. , 2. ],
       [5.6, 2.7, 4.2, 1.3, 2. ],
       [5.7, 3. , 4.2, 1.2, 2. ],
       [5.7, 2.9, 4.2, 1.3, 2. ],
       [6.2, 2.9, 4.3, 1.3, 2. ],
       [5.1, 2.5, 3. , 1.1, 2. ],
       [5.7, 2.8, 4.1, 1.3, 2. ]])

>>> iris3
array([[6.9, 3.1, 5.4, 2.1, 3. ],
       [6.7, 3.1, 5.6, 2.4, 3. ],
       [6.9, 3.1, 5.1, 2.3, 3. ],
       [5.8, 2.7, 5.1, 1.9, 3. ],
       [6.8, 3.2, 5.9, 2.3, 3. ],
       [6.7, 3.3, 5.7, 2.5, 3. ],
       [6.7, 3. , 5.2, 2.3, 3. ],
       [6.3, 2.5, 5. , 1.9, 3. ],
       [6.5, 3. , 5.2, 2. , 3. ],
       [6.2, 3.4, 5.4, 2.3, 3. ]])
```

© NCERT not to be republished

NOTES

```
# Solution to Q6
>>>  header  =np.array(["sepal  length",  "sepal
     width", "petal length", "petal width",
     "Species No"])

# Solution to Q7
>>> print(header)
['sepal length' 'sepal width' 'petal length' 'petal
width' 'Species No']

# Solution to Q8
# Stats for array iris
# Finds the max of the data for sepal length, sepal
width, petal length, petal width, Species No
>>> iris_max = iris.max(axis=0)
>>> iris_max
array([6.9, 3.9, 5.9, 2.5, 3. ])


# Finds the min of the data for sepal length, sepal
# width, petal length, petal width, Species No
>>> iris_min = iris.min(axis=0)
>>> iris_min
array([4.4, 2.3, 1.3, 0.1, 1. ])

# Finds the mean of the data for sepal length, sepal
# width, petal length, petal width, Species No
>>> iris_avg = iris.mean(axis=0).round(2)
>>> iris_avg
array([5.68, 3.03, 3.61, 1.22, 2.  ])

# Finds the standard deviation of the data for sepal
# length, sepal width, petal length, petal width,
# Species No
>>> iris_std = iris.std(axis=0).round(2)
>>> iris_std
array([0.76, 0.35, 1.65, 0.82, 0.82])

# Solution to Q9
>>> iris1_max = iris1.max(axis=0)
>>> iris1_max
array([5.4, 3.9, 1.7, 0.4, 1. ])

>>> iris2_max = iris2.max(axis=0)
>>> iris2_max
array([6.2, 3. , 4.6, 1.4, 2. ])
```

© NCERT not to be republished

NOTES

```
>>> iris3_max = iris3.max(axis=0)
>>> iris3_max
array([6.9, 3.4, 5.9, 2.5, 3. ])

>>> iris1_min = iris1.min(axis=0)
>>> iris1_min
array([4.4, 2.9, 1.3, 0.1, 1. ])
>>> iris2_min = iris2.min(axis=0)
>>> iris2_min
array([5. , 2.3, 3. , 1. , 2. ])

>>> iris3_min = iris3.min(axis=0)
>>> iris3_min
array([5.8, 2.5, 5. , 1.9, 3. ])

>>> iris1_avg = iris1.mean(axis=0)
>>> iris1_avg
array([4.86, 3.31, 1.45, 0.22, 1.  ])

>>> iris2_avg = iris2.mean(axis=0)
>>> iris2_avg
array([5.64, 2.73, 4.03, 1.23, 2.  ])

>>> iris3_avg = iris3.mean(axis=0)
>>> iris3_avg
array([6.55, 3.04, 5.36, 2.2 , 3.  ])

>>> iris1_std = iris1.std(axis=0).round(2)
>>> iris1_std
array([0.28, 0.29, 0.1 , 0.07, 0.  ])

>>> iris2_std = iris2.std(axis=0).round(2)
>>> iris2_std
array([0.36, 0.22, 0.47, 0.11, 0.  ])

>>> iris3_std = iris3.std(axis=0).round(2)
>>> iris3_std
array([0.34, 0.25, 0.28, 0.2 , 0.  ])


# Solution to Q10 (solve other parts on the same lines)
# min sepal length of each species Vs the min sepal
# length in the data set
>>> iris1_min[0] > iris_min[0]  #sepal length
False
```

© NCERT not to be republished

NOTES

```
>>> iris2_min[0] > iris_min[0]
True
>>> iris3_min[0] > iris_min[0]
True

# Solution to Q11
#Compare Iris setosa and Iris virginica
>>> iris1_avg[1] > iris2_avg[1] #sepal width
True

# Solution to Q12
>>> iris1_avg[2] > iris2_avg[2] #petal length
False

# Solution to Q13
>>> iris1_avg[3] > iris2_avg[3] #petal width
False

# Solution to Q14
>>> np.savetxt('C:/NCERT/IrisMeanValues.txt',
    iris_avg, delimiter = ',')

# Solution to Q15
>>> np.savetxt('C:/NCERT/IrisStat.txt', (iris_
    max, iris_avg, iris_min), delimiter=',')
```

© NCERT not to be republished

11149CH07

Chapter 7

Database Concepts

"Inconsistency of your mind... Can damage your memory... Remove the inconsistent data... And keep the original one !!!"

— Nisarga Jain

In this chapter

- Introduction
- File System
- Database Management System
- Relational Data Model
- Keys in a Relational Database

7.1 Introduction

After learning about importance of data in the previous chapter, we need to explore the methods to store and manage data electronically. Let us take an example of a school that maintains data about its students, along with their attendance record and guardian details.

The class teacher marks daily attendance of the students in the attendance register. The teacher records 'P' for present or 'A' for absent against each student's roll number on each working day. If class strength is 50 and total working days in

© NCERT not to be republished

a month are 26, the teacher needs to record 50 × 26 records manually in the register every month. As the volume of data increases, manual data entry becomes tedious. Following are some of the limitations of manual record keeping in this example:

1) Entry of student details (Roll number and name) in the new attendance register when the student is promoted to the next class.
2) Writing student details on each month's attendance page where inconsistency may happen due to incorrectly written names, skipped student records, etc.
3) Loss of data in case attendance register is lost or damaged.
4) Erroneous calculation while consolidating attendance record manually.

The office staff also manually maintain Student details viz. Roll Number, Name and Date of Birth with respective guardian details viz. Guardian name, Contact Number and Address. This is required for correspondence with guardian regarding student attendance and result.

Finding information from a huge volume of papers or deleting/modifying an entry is a difficult task in pen and paper based approach. To overcome the hassles faced in manual record keeping, it is desirable to store attendance record and student details on separate data files on a computerized system, so that office staff and teachers can:

1) Simply copy the student details to the new attendance file from the old attendance file when students are promoted to next class.
2) Find any data about student or guardian.
3) Add more details to existing data whenever a new student joins the school.
4) Modify stored data like details of student or guardian whenever required.
5) Remove/delete data whenever a student leaves the school.

Activity 7.1

Visit a few shops where records are maintained manually and identify a few limitations of manual record keeping faced by them.

7.2 FILE SYSTEM

A file can be understood as a container to store data in a computer. Files can be stored on the storage device of a computer system. Contents of a file can be texts, computer program code, comma separated values

© NCERT not to be republished

(CSV), etc. Likewise, pictures, audios/videos, web pages are also files.

Files stored on a computer can be accessed directly and searched for desired data. But to access data of a file through software, for example, to display monthly attendance report on school website, one has to write computer programs to access data from files.

Continuing the example of attendance at school, we need to store data about students and attendance in two separate files. Table 7.1 shows the contents of STUDENT file which has six columns, as detailed below:

RollNumber – Roll number of the student

SName – Name of the student

SDateofBirth – Date of birth of the student

GName – Name of the guardian

GPhone – Phone number of the student guardian

GAddress – Address of the guardian of the student

Table 7.1 STUDENT file maintained by office staff

Roll Number	SName	SDateof Birth	GName	GPhone	GAddress
1	Atharv Ahuja	2003-05-15	Amit Ahuja	5711492685	G-35, Ashok Vihar, Delhi
2	Daizy Bhutia	2002-02-28	Baichung Bhutia	3612967082	Flat no. 5, Darjeeling Appt., Shimla
3	Taleem Shah	2002-02-28	Himanshu Shah	4726309212	26/77, West Patel Nagar, Ahmedabad
4	John Dsouza	2003-08-18	Danny Dsouza		S -13, Ashok Village, Daman
5	Ali Shah	2003-07-05	Himanshu Shah	4726309212	26/77, West Patel Nagar, Ahmedabad
6	Manika P.	2002-03-10	Sujata P.	3801923168	HNO-13, B- block, Preet Vihar, Madurai

Table 7.2 shows another file called ATTENDANCE which has four columns, as detailed below:

AttendanceDate – Date for which attendance was marked

RollNumber – Roll number of the student

SName – Name of the student

AttendanceStatus – Marked as P (present) or A (absent)

© NCERT not to be republished

Table 7.2 ATTENDANCE file maintained by class teacher

AttendanceDate	RollNumber	SName	AttendanceStatus
2018-09-01	1	Atharv Ahuja	P
2018-09-01	2	Daizy Bhutia	P
2018-09-01	3	Taleem Shah	A
2018-09-01	4	John Dsouza	P
2018-09-01	5	Ali Shah	A
2018-09-01	6	Manika P.	P
2018-09-02	1	Atharv Ahuja	P
2018-09-02	2	Daizy Bhutia	P
2018-09-02	3	Taleem Shah	A
2018-09-02	4	John Dsouza	A
2018-09-02	5	Ali Shah	P
2018-09-02	6	Manika P.	P

7.2.1 Limitations of a File System

File system becomes difficult to handle when number of files increases and volume of data also grows. Following are some of the limitations of file system:

(A) Difficulty in Access

Files themselves do not provide any mechanism to retrieve data. Data maintained in a file system are accessed through application programs. While writing such programs, the developer may not anticipate all the possible ways in which data may be accessed. So, sometimes it is difficult to access data in the required format and one has to write application program to access data.

(B) Data Redundancy

Redundancy means same data are duplicated in different places (files). In our example, student names are maintained in both the files. Besides, in Table 7.1, students with roll numbers 3 and 5 have same guardian name and therefore same guardian name is maintained twice. Both these are examples of redundancy which is difficult to avoid in a file system. Redundancy leads to excess storage use and may cause data inconsistency also.

(C) Data Inconsistency

Data inconsistency occurs when same data maintained in different places do not match. If a student wants to get changed the spelling of her name, it needs to be

© NCERT not to be republished

Notes

changed in SName column in both the files. Likewise, if a student leaves school, the details need to be deleted from both the files. As the files are being maintained by different people, the changes may not happen in one of the files. In that case, the student name will be different (inconsistent) in both the files.

(D) Data Isolation

Both the files presented at Table 7.1 (STUDENT) and at Table 7.2 (ATTENDANCE) are related to students. But there is no link or mapping between them. The school will have to write separate programs to access these two files. This is because data mapping is not supported in file system. In a more complex system where data files are generated by different person at different times, files being created in isolation may be of different formats. In such case, it is difficult to write new application programs to retrieve data from different files maintained at multiple places, as one has to understand the underlying structure of each file as well.

(E) Data Dependence

Data are stored in a specific format or structure in a file. If the structure or format itself is changed, all the existing application programs accessing that file also need to be change. Otherwise, the programs may not work correctly. This is data dependency. Hence, updating the structure of a data file requires modification in all the application programs accessing that file.

(F) Controlled Data Sharing

There can be different category of users like teacher, office staff and parents. Ideally, not every user should be able to access all the data. As an example, guardians and office staff can only see the student attendance data but should not be able to modify/delete it. It means these users should be given limited access (read only) to the ATTENDANCE file. Only the teacher should be able to update the attendance data. It is very difficult to enforce this kind of access control in a file system while accessing files through application programs.

7.3 Database Management System

Limitations faced in file system can be overcome by storing the data in a database where data are logically related. We can organise related data in a database so that it can be managed in an efficient and easy way.

© NCERT not to be republished

A database management system (DBMS) or database system in short, is a software that can be used to create and manage databases. DBMS lets users to create a database, store, manage, update/modify and retrieve data from that database by users or application programs. Some examples of open source and commercial DBMS include MySQL, Oracle, PostgreSQL, SQL Server, Microsoft Access, MongoDB.

A database system hides certain details about how data are actually stored and maintained. Thus, it provides users with an abstract view of the data. A database system has a set of programs through which users or other programs can access, modify and retrieve the stored data.

The DBMS serves as an interface between the database and end users or application programs. Retrieving data from a database through special type of commands is called querying the database. In addition, users can modify the structure of the database itself through a DBMS.

Some database management systems include a graphical user interface for users to create and manage databases. Other database systems use a command line interface that requires users to use programming commands to create and manage databases.

Databases are widely used in various fields. Some applications are given in Table 7.3.

Table 7.3 Use of Database in Real-life Applications

Application	Database to maintain data about
Banking	customer information, account details, loan details, transaction details, etc.
Crop Loan	kisan credit card data, farmer's personal data, land area and cultivation data, loan history, repayment data, etc.
Inventory Management	product details, customer information, order details, delivery data, etc.
Organisation Resource Management	employee records, salary details, department information, branch locations, etc.
Online Shopping	items description, user login details, users preferences details, etc.

7.3.1 File System to DBMS

Let us revisit our school example where two data files were maintained (Table 7.1 by office and Table 7.2 by teacher). Let us now design a database to store data of those two files. We know that tables in a database are linked or related through one or more common columns or fields. In our example, the STUDENT (Table 7.1) file and ATTENDANCE (Table 7.2) file have RollNumber and SName as common field names. In order to convert

© NCERT not to be republished

these two files into a database, we need to incorporate the following changes:

a) SName need not be maintained in ATTENDANCE file as it is already there in STUDENT. Details for a student can be retrieved through the common field RollNumber in both the files.

b) If two siblings are in the same class, then same guardian details (GName, GPhone and GAddress) are maintained for both the siblings. We know this is a redundancy and by using a database we can avoid this. So let us split the STUDENT file into two file (STUDENT file and GUARDIAN) file so that each guardian data are maintained only once.

c) One and more guardians can have the same name. So it will not be possible to identify which guardian is related to which student. In such case, we need to create an additional column, say GUID (Guardian ID) that will take unique value for each record in the GUARDIAN file. The column GUID will also be kept with STUDENT file for relating these two files.

High Cost is incurred while shifting from file system to DBMS:

- Purchasing sophisticated hardware and software.
- Training users for querying.
- Recurrent cost to take regular backup and perform recovery operations.

Note: We could distinguish guardians by their phone numbers also. But, phone number can change, and therefore may not truly distinguish guardian.

Figure 7.1 shows the related data files for the STUDENT, GUARDIAN and ATTENDANCE details. Note that this is not the complete database schema since it does not show any relationship among tables.

STUDENT	GUARDIAN	ATTENDANCE
RollNumber SName SDateofBirth GUID	GUID GName GPhone GAddress	AttendanceDate RollNumber AttendanceStatus

Figure 7.1: Record structure of three files in STUDENTATTENDANCE Database

The tables shown at Figure 7.1 are empty, which are to be populated with actual data as shown in Table 7.4, 7.5 and 7.6.

Table 7.4 Snapshot of STUDENT table

RollNumber	SName	SDateofBirth	GUID
1	Atharv Ahuja	2003-05-15	444444444444
2	Daizy Bhutia	2002-02-28	111111111111

© NCERT not to be republished

3	Taleem Shah	2002-02-28	
4	John Dsouza	2003-08-18	333333333333
5	Ali Shah	2003-07-05	101010101010
6	Manika P.	2002-03-10	466444444666

Table 7.5 Snapshot of GUARDIAN table

GUID	GName	GPhone	GAddress
444444444444	Amit Ahuja	5711492685	G-35, Ashok Vihar, Delhi
111111111111	Baichung Bhutia	3612967082	Flat no. 5, Darjeeling Appt., Shimla
101010101010	Himanshu Shah	4726309212	26/77, West Patel Nagar, Ahmedabad
333333333333	Danny Dsouza		S -13, Ashok Village, Daman
466444444666	Sujata P.	3801923168	HNO-13, B- block, Preet Vihar, Madurai

Table 7.6 Snapshot of ATTENDANCE table

Date	RollNumber	Status
2018-09-01	1	P
2018-09-01	2	P
2018-09-01	3	A
2018-09-01	4	P
2018-09-01	5	A
2018-09-01	6	P
2018-09-02	1	P
2018-09-02	2	P
2018-09-02	3	A
2018-09-02	4	A
2018-09-02	5	P
2018-09-02	6	P

Figure 7.2 shows a simplified database called STUDENTATTENDANCE, which is used to maintain data about the student, guardian and attendance. As shown here, the DBMS maintains a single repository of data at a centralized location and can be used by multiple users (office staff, teacher) at the same time.

7.3.2 Key Concepts in DBMS

In order to efficiently manage data using a DBMS, let us understand certain key terms:

(A) Database Schema

Database Schema is the design of a database. It is the skeleton of the database that represents the structure (table names and their fields/columns), the type of data each column can hold, constraints on the data to be stored (if any), and the relationships among the tables.

© NCERT not to be republished

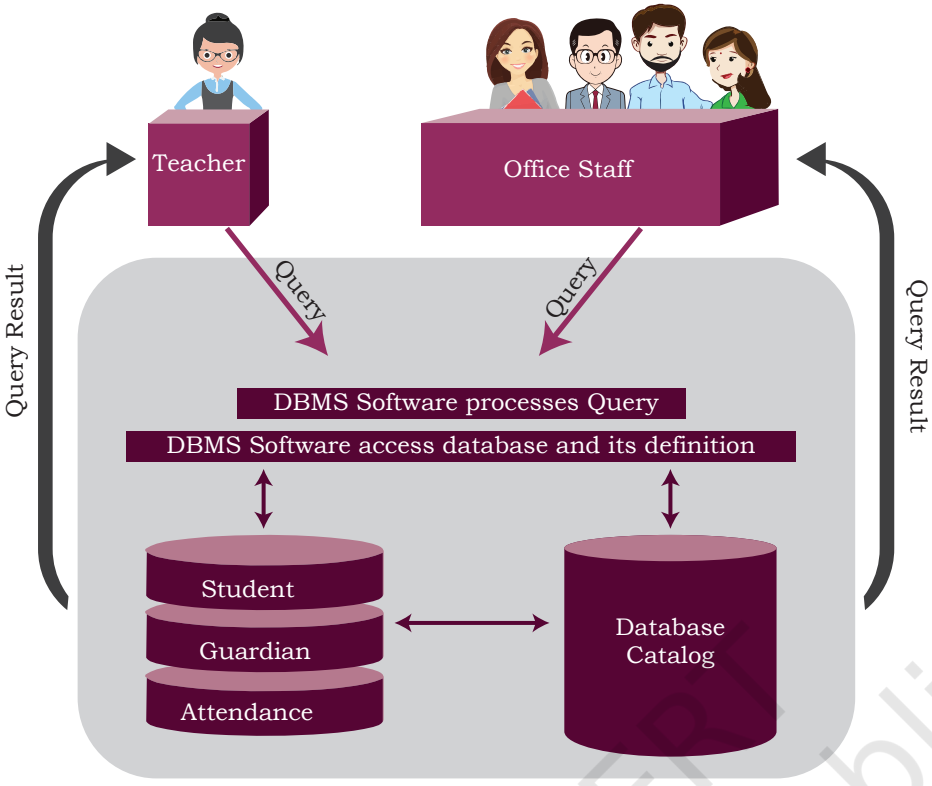

Figure 7.3: StudentAttendance Database Environment

Database schema is also called the visual or logical architecture as it tells us how the data are organised in a database.

(B) Data Constraint

Sometimes we put certain restrictions or limitations on the type of data that can be inserted in one or more columns of a table. This is done by specifying one or more constraints on that column(s) while creating the tables. For example, one can define the constraint that the column *mobile number* can only have non-negative integer values of exactly 10 digits. Since each student shall have one unique roll number, we can put the NOT NULL and UNIQUE constraints on the RollNumber column. Constraints are used to ensure accuracy and reliability of data in the database

(C) Meta-data or Data Dictionary

The database schema along with various constraints on the data is stored by DBMS in a database catalog or dictionary, called meta-data. A meta-data is data about the data.

(D) Database Instance

When we define database structure or schema, state of database is empty i.e. no data entry is there. After

© NCERT not to be republished

loading data, the state or snapshot of the database at any given time is the database instance. We may then retrieve data through queries or manipulate data through updation, modification or deletion. Thus, the state of database can change, and thus a database schema can have many instances at different times.

(E) Query

A query is a request to a database for obtaining information in a desired way. Query can be made to get data from one table or from a combination of tables. For example, "find names of all those students present on Attendance Date 2000-01-02" is a query to the database. To retrieve or manipulate data, the user needs to write query using a query language called, which is discussed in chapter 8.

Limitations of DBMS

Increased Complexity: Use of DBMS increases the complexity of maintaining functionalities like security, consistency, sharing and integrity

Increased data vulnerability: As data are stored centrally, it increases the chances of loss of data due to any failure of hardware or software. It can bring all operations to a halt for all the users.

(F) Data Manipulation

Modification of database consists of three operations viz. Insertion, Deletion or Update. Suppose Rivaan joins as a new student in the class then the student details need to be added in STUDENT as well as in GUARDIAN files of the Student Attendance database. This is called Insertion operation on the database. In case a student leaves the school, then his/her data as well as her guardian details need to be removed from STUDENT, GUARDIAN and ATTENDANCE files, respectively. This is called Deletion operation on the database. Suppose Atharv's Guardian has changed his mobile number, his GPhone should be updated in GUARDIAN file. This is called Update operation on the database.

(G) Database Engine

Database engine is the underlying component or set of programs used by a DBMS to create database and handle various queries for data retrieval and manipulation.

7.4 RELATIONAL DATA MODEL

Different types of DBMS are available and their classification is done based on the underlying data model. A data model describes the structure of the database, including how data are defined and represented, relationships among data, and the constraints. The most commonly used data model is Relational Data Model. Other types of data models include object-oriented data model, entity-relationship data model, document model and hierarchical data model. This book discusses the DBMS based on relational data model.

© NCERT not to be republished

In relational model, tables are called relations that store data for different columns. Each table can have multiple columns where each column name should be unique. For example, each row in the table represents a related set of values. Each row of Table 7.5 represents a particular guardian and has related values viz. guardian's ID with guardian name, address and phone number. Thus, a table consists of a collection of relationships.

It is important to note here that relations in a database are not independent tables, but are associated with each other. For example, relation ATTENDANCE has attribute RollNumber which links it with corresponding student record in relation STUDENT. Similarly, attribute GUID is placed with STUDENT table for extracting guardian details of a particular student. If linking attributes are not there in appropriate relations, it will not be possible to keep the database in correct state and retrieve valid information from the database.

Figure 7.3 shows the relational database Student Attendance along with the three relations (tables) STUDENT, ATTENDANCE and GUARDIAN.

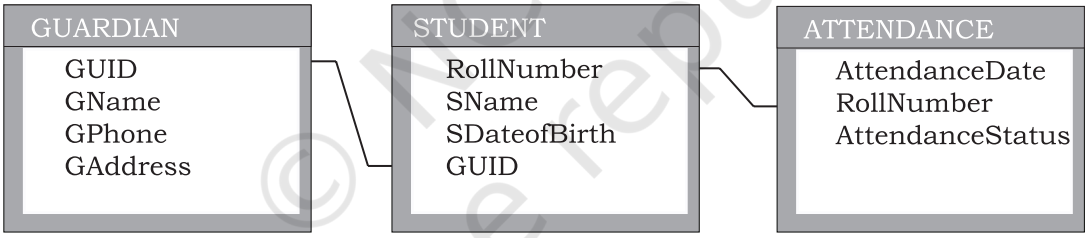

Figure 7.4: Representing StudentAttendance Database using Relational Data Model

Table 7.7 Relation schemas along with its description of Student Attendance database

Relation Scheme	Description of attributes
STUDENT(RollNumber, SName, SDateofBirth, GUID)	RollNumber: unique id of the student SName: name of the student SDateofBirth: date of birth of the student GUID: unique id of the guardian of the student
ATTENDANCE (AttendanceDate, RollNumber, AttendanceStatus)	AttendanceDate: date on which attendance is taken RollNumber: roll number of the student AttendanceStatus: whether present (P) or absent(A) Note that combination of AttendanceDate and RollNumber will be unique in each record of the table
GUARDIAN(GUID, GName, GPhone, GAddress)	GUID: unique id of the guardian GName: name of the guardian GPhone: contact number of the guardian GAddress: contact address of the guardian

© NCERT not to be republished

Each tuple (row) in a relation (table) corresponds to data of a real world entity (for example, Student, Guardian, and Attendance). In the GUARDIAN relation (Table 7.5), each row represents the facts about the guardian and each column name in the GUARDIAN table is used to interpret the meaning of data stored under that column. A database that is modeled on relational data model concept is called Relational Database. Figure 7.4 shows relation GUARDIAN with some populated data.

Let us now understand the commonly used terminologies in relational data model using Figure 7.4.

Relation GUARDIAN with 4 attribute/ columns

GUID	GName	GPhone	GAddress
444444444444	Amit Ahuja	5711492685	G-35, Ashok Vihar, Delhi
111111111111	Baichung Bhutia	3612967082	Flat no. 5, Darjeeling Appt., Shimla
101010101010	Himanshu Shah	4726309212	26/77, West Patel Nagar, Ahmedabad
333333333333	Danny Dsouza		S -13, Ashok Village, Daman
466444444666	Sujata P.	3801923168	HNO-13, B- block, Preet Vihar, Madurai

Relation State

Record/tuple/row

Facts about RELATION GUARDIAN:
1. Degree (Number of attributes) = 4
2. Cardinality (Number of rows/tuples/records) = 5
3. Relation is a flat file i.e, each column has a single value and each record has same number of columns

Figure 7.5: Relation GUARDIAN with its Attributes and Tuples

i) **ATTRIBUTE:** Characteristic or parameters for which data are to be stored in a relation. Simply stated, the columns of a relation are the attributes which are also referred as fields. For example, GUID, GName, GPhone and GAddress are attributes of relation GUARDIAN.

ii) **TUPLE:** Each row of data in a relation (table) is called a tuple. In a table with n columns, a tuple is a relationship between the n related values.

iii) **DOMAIN:** It is a set of values from which an attribute can take a value in each row. Usually, a data type is used to specify domain for an attribute. For example, in STUDENT relation, the attribute RollNumber takes integer values and hence its domain is a set of integer values. Similarly, the set of character strings constitutes the domain of the attribute SName.

© NCERT not to be republished

Notes

iv) **DEGREE:** The number of attributes in a relation is called the Degree of the relation. For example, relation GUARDIAN with four attributes is a relation of degree 4.

v) **CARDINALITY:** The number of tuples in a relation is called the Cardinality of the relation. For example, the cardinality of relation GUARDIAN is 5 as there are 5 tuples in the table.

7.4.1 Three Important Properties of a Relation

In relational data model, following three properties are observed with respect to a relation which makes a relation different from a data file or a simple table.

Property 1: imposes following rules on an attribute of the relation.

- Each attribute in a relation has a unique name.
- Sequence of attributes in a relation is immaterial.

Property 2: governs following rules on a tuple of a relation.

- Each tuple in a relation is distinct. For example, data values in no two tuples of relation ATTENDANCE can be identical for all the attributes. Thus, each tuple of a relation must be uniquely identified by its contents.
- Sequence of tuples in a relation is immaterial. The tuples are not considered to be ordered, even though they appear to be in tabular form.

Property 3: imposes following rules on the state of a relation.

- All data values in an attribute must be from the same domain (same data type).
- Each data value associated with an attribute must be atomic (cannot be further divisible into meaningful subparts). For example, GPhone of relation GUARDIAN has ten digit numbers which is indivisible.
- No attribute can have many data values in one tuple. For example, Guardian cannot specify multiple contact numbers under GPhone attribute.
- A special value "NULL" is used to represent values that are unknown or non-applicable to certain attributes. For example, if a guardian does not share his or her contact number with the school authorities, then GPhone is set to NULL (data unknown).

© NCERT not to be republished

NOTES

7.5 KEYS IN A RELATIONAL DATABASE

The tuples within a relation must be distinct. It means no two tuples in a table should have same value for all attributes. That is, there should be at least one attribute in which data are distinct (unique) and not NULL. That way, we can uniquely distinguish each tuple of a relation. So, relational data model imposes some restrictions or constraints on the values of the attributes and how the contents of one relation be referred through another relation. These restrictions are specified at the time of defining the database through different types of keys as given below:

7.5.1 Candidate Key

A relation can have one or more attributes that takes distinct values. Any of these attributes can be used to uniquely identify the tuples in the relation. Such attributes are called candidate keys as each of them are candidates for the primary key.

As shown in Figure 7.4, the relation GUARDIAN has four attributes out of which GUID and GPhone always take unique values. No two guardians will have same phone number or same GUID. Hence, these two attributes are the candidate keys as they both are candidates for primary key.

7.5.2 Primary Key

Out of one or more candidate keys, the attribute chosen by the database designer to uniquely identify the tuples in a relation is called the primary key of that relation. The remaining attributes in the list of candidate keys are called the alternate keys.

In the relation GUARDIAN, suppose GUID is chosen as primary key, then GPhone will be called the alternate key.

7.5.3 Composite Primary Key

If no single attribute in a relation is able to uniquely distinguish the tuples, then more than one attribute are taken together as primary key. Such primary key consisting of more than one attribute is called Composite Primary key.

In relation ATTENDANCE, Roll Number cannot be used as primary key as roll number of same student will appear in another row for a different date. Similarly, in relation Attendance, AttendanceDate cannot be used as primary key because same date is repeated for each roll number. However combination of these two attributes RollNumber and AttendanceDate together would always have unique value in ATTENDANCE table as on any working day, of a student would be marked attendance only once. Hence {RollNumber,

© NCERT not to be republished

AttendanceDate} will make the of ATTENDANCE relation composite primary key.

7.5.4 Foreign Key

A foreign key is used to represent the relationship between two relations. A foreign key is an attribute whose value is derived from the primary key of another relation. This means that any attribute of a relation (referencing), which is used to refer contents from another (referenced) relation, becomes foreign key if it refers to the primary key of referenced relation. The referencing relation is called Foreign Relation. In some cases, foreign key can take NULL value if it is not the part of primary key of the foreign table. The relation in which the referenced primary key is defined is called primary relation or master relation.

In Figure 7.5, two foreign keys in Student Attendance database are shown using schema diagram where the foreign key is displayed as a directed arc (arrow) originating from it and ending at the corresponding attribute of the primary key of the referenced table. The underlined attributes make the primary key of that table.

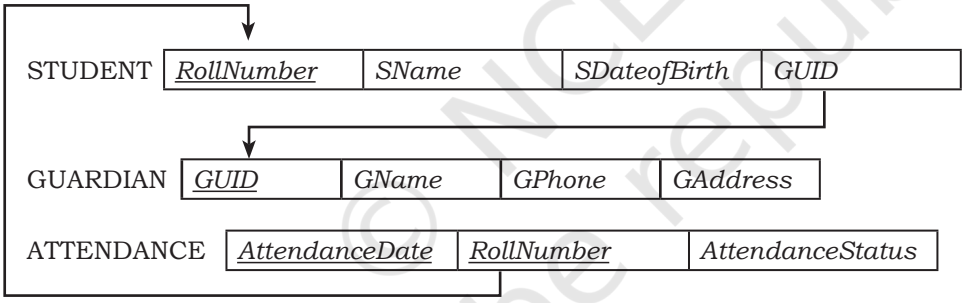

Figure 7.2: StudentAttendance Database with the Primary and Foreign keys

SUMMARY

- A file in a file system is a container to store data in a computer.
- File system suffers from Data Redundancy, Data Inconsistency, Data Isolation, Data Dependence and Controlled Data sharing.
- Database Management System (DBMS) is a software to create and manage databases. A database is a collection of tables.
- Database schema is the design of a database
- A database constraint is a restriction on the type of data that that can be inserted into the table.
- Database schema and database constraints are stored in database Catalog.

© NCERT not to be republished

- Whereas the snapshot of the database at any given time is the database instance.
- A query is a request to a database for information retrieval and data manipulation (insertion, deletion or update). It is written in Structured Query Language (SQL).
- Relational DBMS (RDBMS) is used to store data in related tables. Rows and columns of a table are called tuples and attributed respectively. A table is referred to as a relation.
- Destructions on data stored in a RDBMS is applied by use of keys such as Candidate Key, Primary Key, Composite Primary Key, Foreign Key.
- Primary key in a relation is used for unique identification of tuples.
- Foreign key is used to relate two tables or relations.
- Each column in a table represents a feature (attribute) of a record. Table stores the information for an entity whereas a row represents a record.
- Each row in a table represents a record. A tuple is a collection of attribute values that makes a record unique.
- A tuple is a unique entity whereas attribute values can be duplicate in the table.
- SQL is the standard language for RDBMS systems like MySQL.

EXERCISE

1. Give the terms for each of the following:
   a) Collection of logically related records.
   b) DBMS creates a file that contains description about the data stored in the database.
   c) Attribute that can uniquely identify the tuples in a relation.
   d) Special value that is stored when actual data value is unknown for an attribute.
   e) An attribute which can uniquely identify tuples of the table but is not defined as primary key of the table.
   f) Software that is used to create, manipulate and maintain a relational database.
2. Why foreign keys are allowed to have NULL values? Explain with an example.

© NCERT not to be republished

Notes

3. Differentiate between:
   a) Database state and database schema
   b) Primary key and foreign key
   c) Degree and cardinality of a relation
4. Compared to a file system, how does a database management system avoid redundancy in data through a database?
5. What are the limitations of file system that can be overcome by a relational DBMS?
6. A school has a rule that each student must participate in a sports activity. So each one should give only one preference for sports activity. Suppose there are five students in a class, each having a unique roll number. The class representative has prepared a list of sports preferences as shown below. Answer the following:

Table: Sports Preferences

Roll_no	Preference
9	Cricket
13	Football
17	Badminton
17	Football
21	Hockey
24	NULL
NULL	Kabaddi

   a) Roll no 24 may not be interested in sports. Can a NULL value be assigned to that student's preference field?
   b) Roll no 17 has given two preferences sports. Which property of relational DBMC is violated here? Can we use any constraint or key in the relational DBMS to check against such violation, if any?
   c) Kabaddi was not chosen by any student. Is it possible to have this tuple in the Sports Preferences relation?
7. In another class having 2 sections, the two respective class representatives have prepared 2 separate Sports Preferences tables, as shown below:

   Sports preference of section 1 (arranged on roll number column)

Table: Sports Preferences

Roll_no	Sports
9	Cricket
13	Football
17	Badminton
21	Hockey
24	Cricket

   Sports preference of section 2 (arranged on Sports name column, and column order is also different)

© NCERT not to be republished

NOTES

Table: Sports Preferences

Sports	Roll_no
Badminton	17
Cricket	9
Cricket	24
Football	13
Hockey	21

Are the states of both the relations equivalent? Justify.

8. The school canteen wants to maintain records of items available in the school canteen and generate bills when students purchase any item from the canteen. The school wants to create a canteen database to keep track of items in the canteen and the items purchased by students. Design a database by answering the following questions:
   a) To store each item name along with its price, what relation should be used? Decide appropriate attribute names along with their data type. Each item and its price should be stored only once. What restriction should be used while defining the relation?
   b) In order to generate bill, we should know the quantity of an item purchased. Should this information be in a new relation or a part of the previous relation? If a new relation is required, decide appropriate name and data type for attributes. Also, identify appropriate primary key and foreign key so that the following two restrictions are satisfied:
      i) The same bill cannot be generated for different orders.
      ii) Bill can be generated only for available items in the canteen.
   c) The school wants to find out how many calories students intake when they order an item. In which relation should the attribute 'calories' be stored?

9. An organisation wants to create a database EMP-DEPENDENT to maintain following details about its employees and their dependent.

```
EMPLOYEE(AadharNumber, Name, Address,
        Department,EmployeeID)

DEPENDENT(EmployeeID, DependentName,
        Relationship)
```

   a) Name the attributes of EMPLOYEE, which can be used as candidate keys.
   b) The company wants to retrieve details of dependent of a particular employee. Name the tables and the key which are required to retrieve this detail.

© NCERT not to be republished

c) What is the degree of EMPLOYEE and DEPENDENT relation?

10. School uniform is available at M/s Sheetal Private Limited. They have maintained SCHOOL_UNIFORM Database with two relations viz. UNIFORM and COST. The following figure shows database schema and its state.

School Uniform Database

Attributes and Constraints

Table: UNIFORM			
Attribute	**UCode**	**UName**	**UColor**
Constraints	Primary Key	Not Null	-

Table: COST			
Attribute	**UCode**	**Size**	**Price**
Constraints	Composite Primary Key		>0

Table: UNIFORM

UCode	UName	UColor
1	Shirt	White
2	Pant	Grey
3	Skirt	Grey
4	Tie	Blue
5	Socks	Blue
6	Belt	Blue

Table:

UCode	Size	COST Price
1	M	500
1	L	580
1	XL	620
2	M	810
2	L	890
2	XL	940
3	M	770
3	L	830
3	XL	910
4	S	150
4	L	170
5	S	180
5	L	210
6	M	110
6	L	140
6	XL	160

a) Can they insert the following tuples to the UNIFORM Relation? Give reasons in support of your answer.
   i) 7, Handkerchief, NULL
   ii) 4, Ribbon, Red
   iii) 8, NULL, White

b) Can they insert the following tuples to the COST Relation? Give reasons in support of your answer.
   i) 7, S, 0
   ii) 9, XL, 100

11. In a multiplex, movies are screened in different auditoriums. One movie can be shown in more than one auditorium. In order to maintain the record of movies, the multiplex maintains a relational database consisting of two relations viz. MOVIE and AUDI respectively as shown below:

```
Movie(Movie_ID, MovieName, ReleaseDate)
Audi(AudiNo, Movie_ID, Seats, ScreenType,
                    TicketPrice)
```

© NCERT not to be republished

a) Is it correct to assign Movie_ID as the primary key in the MOVIE relation? If no, then suggest an appropriate primary key.
b) Is it correct to assign AudiNo as the primary key in the AUDI relation? If no, then suggest appropriate primary key.
c) Is there any foreign key in any of these relations?

Student Project Database

Table: STUDENT

Roll No	Name	Class	Section	Registration_ID
11	Mohan	XI	1	IP-101-15
12	Sohan	XI	2	IP-104-15
21	John	XII	1	CS-103-14
22	Meena	XII	2	CS-101-14
23	Juhi	XII	2	CS-101-10

Table: PROJECT

ProjectNo	PName	SubmissionDate
101	Airline Database	12/01/2018
102	Library Database	12/01/2018
103	Employee Database	15/01/2018
104	Student Database	12/01/2018
105	Inventory Database	15/01/2018
106	Railway Database	15/01/2018

Table: PROJECT ASSIGNED

Registration_ID	ProjectNo
IP-101-15	101
IP-104-15	103
CS-103-14	102
CS-101-14	105
CS-101-10	104

12. For the above given database STUDENT-PROJECT, answer the following:
    a) Name primary key of each table.
    b) Find foreign key(s) in table PROJECT-ASSIGNED.
    c) Is there any alternate key in table STUDENT? Give justification for your answer.
    d) Can a user assign duplicate value to the field RollNo of STUDENT table? Jusify.
13. For the above given database STUDENT-PROJECT, can we perform the following operations?
    a) Insert a student record with missing roll number value.
    b) Insert a student record with missing registration number value.
    c) Insert a project detail without submission-date.
    d) Insert a record with registration ID IP-101-19 and ProjectNo 206 in table PROJECT-ASSIGNED.

© NCERT not to be republished

11149CH08

Introduction to Structured Query Language (SQL)

Chapter 8

"The most important motivation for the research work that resulted in the relational model was the objective of providing a sharp and clear boundary between the logical and physical aspects of database management."

– E. F. Codd

In this chapter

- Introduction
- Structured Query Language (SQL)
- Data Types and Constraints in MySQL
- SQL for Data Definition
- SQL for Data Manipulation
- SQL for Data Query
- Data Updation and Deletion

8.1 Introduction

We have learnt about Relational Database Management System (RDBMS) and purpose in the previous chapter. There are many RDBMS such as MySQL, Microsoft SQL Server, PostgreSQL, Oracle, etc. that allow us to create a database consisting of relations and to link one or more relations for efficient querying to store, retrieve and manipulate data on that database. In this chapter, we will learn how to create, populate and query database using MySQL.

© NCERT not to be republished

8.2 STRUCTURED QUERY LANGUAGE (SQL)

One has to write application programs to access data in case of a file system. However, for database management systems there are special kind of programming languages called query language that can be used to access data from the database. The Structured Query Language (SQL) is the most popular query language used by major relational database management systems such as MySQL, ORACLE, SQL Server, etc.

SQL is easy to learn as the statements comprise of descriptive English words and are not case sensitive. We can create and interact with a database using SQL in an efficient and easy way. The benefit with SQL is that we don't have to specify how to get the data from the database. Rather, we simply specify what is to be retrieved, and SQL does the rest. Although called a query language, SQL can do much more besides querying. SQL provides statements for defining the structure of the data, manipulating data in the database, declare constraints and retrieve data from the database in various ways, depending on our requirements.

In this chapter, we will learn how to create a database using MySQL as the RDBMS software. We will create a database called `StudentAttendance` (Figure 7.5) that we had identified in the previous chapter. We will also learn how to populate database with data, manipulate data in that and retrieve data from the database through SQL queries.

8.2.1 Installing MySQL

MySQL is an open source RDBMS software which can be easily downloaded from the official website https://dev.mysql.com/downloads. After installing MySQL, start MySQL service. The appearance of mysql> prompt (Figure 8.1) means that MySQL is ready for us to enter SQL statements.

Few rules to follow while writing SQL statements in MySQL:

- SQL is case insensitive. That means name and NAME are same for SQL.
- Always end SQL statements with a semicolon (;).
- To enter multiline SQL statements, we don't write ';' after the first line. We put enter to continue on next line. The prompt mysql> then changes to '->',

Activity 8.1

Explore LibreOffice Base and compare it with MySQL

© NCERT not to be republished

indicating that statement is continued to the next line. After the last line, put ';' and press enter.

8.3 Data Types and Constraints in MySQL

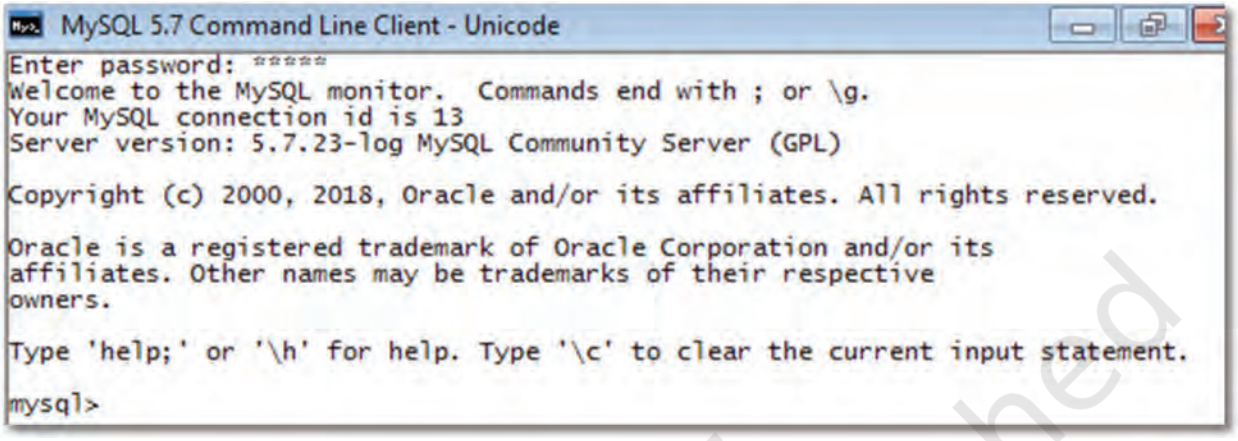

Figure 8.1: MySQL Shell

We know that a database consists of one or more relations and each relation (table) is made up of attributes (column). Each attribute has a data type. We can also specify constraints for each attribute of a relation.

Activity 8.2

What are the other data types supported in MySQL? Are there other variants of integer and float data type?

8.3.1 Data type of Attribute

Data type indicates the type of data value that an attribute can have. The data type of an attribute decides the operations that can be performed on the data of that attribute. For example, arithmetic operations can be performed on numeric data but not on character data. Commonly used data types in MySQL are numeric types, date and time types, and string (character and byte) types as shown in Table 8.1.

Think and Reflect

Can you think of an attribute for which fixed length string is suitable?

Table 8.1 Commonly used data types in MySQL

Data type	Description
CHAR(n)	Specifies character type data of length n where n could be any value from 0 to 255. CHAR is of fixed length, means, declaring CHAR (10) implies to reserve spaces for 10 characters. If data does not have 10 characters (for example, 'city' has four characters), MySQL fills the remaining 6 characters with spaces padded on the right.
VARCHAR(n)	Specifies character type data of length 'n' where n could be any value from 0 to 65535. But unlike CHAR, VARCHAR is a variable-length data type. That is, declaring VARCHAR (30) means a maximum of 30 characters can be stored but the actual allocated bytes will depend on the length of entered string. So 'city' in VARCHAR (30) will occupy the space needed to store 4 characters only.

© NCERT not to be republished

INT	INT specifies an integer value. Each INT value occupies 4 bytes of storage. The range of values allowed in integer type are -2147483648 to 2147483647. For values larger than that, we have to use BIGINT, which occupies 8 bytes.
FLOAT	Holds numbers with decimal points. Each FLOAT value occupies 4 bytes.
DATE	The DATE type is used for dates in 'YYYY-MM-DD' format. YYYY is the 4 digit year, MM is the 2 digit month and DD is the 2 digit date. The supported range is '1000-01-01' to '9999-12-31'.

Think and Reflect

Which two constraints when applied together will produce a Primary Key constraint?

8.3.2 Constraints

Constraints are certain types of restrictions on the data values that an attribute can have. They are used to ensure the accuracy and reliability of data. However, it is not mandatory to define constraint for each attribute of a table. Table 8.2 lists various SQL constraints.

Table 8.2 Commonly used SQL Constraints

Constraint	Description
NOT NULL	Ensures that a column cannot have NULL values where NULL means missing/ unknown/not applicable value.
UNIQUE	Ensures that all the values in a column are distinct/unique.
DEFAULT	A default value specified for the column if no value is provided.
PRIMARY KEY	The column which can uniquely identify each row or record in a table.
FOREIGN KEY	The column which refers to value of an attribute defined as primary key in another table.

8.4 SQL FOR DATA DEFINITION

SQL provides commands for defining the relation schemas, modifying relation schemas and deleting relations. These are called Data Definition Language (DDL) through which the set of relations are specified, including their schema, data type for each attribute, the constraints as well as the security and access related authorisations.

Data definition starts with the create statement. This statement is used to create a database and its tables (relations). Before creating a database, we should be clear about the number of tables in the database, the columns (attributes) in each table along with the data type of each column. This is how we decide the relation schema.

8.4.1 CREATE Database

To create a database, we use the CREATE DATABASE statement as shown in the following syntax:

```
CREATE DATABASE databasename;
```

© NCERT not to be republished

To create a database called StudentAttendance, we will type following command at mysql prompt.

```
mysql> CREATE DATABASE StudentAttendance;
Query OK, 1 row affected (0.02 sec)
```

Note: In LINUX environment, names for database and tables are case-sensitive whereas in WINDOWS, there is no such differentiation. However, as a good practice, it is suggested to write database or table name in the same letter cases that were used at the time of their creation.

A DBMS can manage multiple databases on one computer. Therefore, we need to select the database that we want to use. Once the database is selected, we can proceed with creating tables or querying data. Write the following SQL statement for using the database:

```
mysql> USE StudentAttendance;
Database changed
```

Activity 8.3

Type the statement show database;. Does it show the name of StudentAttendance database?

Initially, the created database is empty. It can be checked by using the Show tables command that lists names of all the tables within a database.

```
mysql> SHOW TABLES;
Empty set (0.06 sec)
```

8.4.2 CREATE Table

After creating database StudentAttendance, we need to define relations (create tables) in this database and specify attributes for each relation along with data types for each attribute. This is done using the CREATE TABLE statement.

Syntax:

```
CREATE TABLE tablename(
attributename1 datatype constraint,
attributename2 datatype constraint,
:
attributenameN datatype constraint);
```

It is important to observe the following points with respect to the Create Table statement:

- N is the degree of the relation, means there are N columns in the table.
- Attribute name specifies the name of the column in the table.
- Datatype specifies the type of data that an attribute can hold.
- Constraint indicates the restrictions imposed on the values of an attribute. By default, each attribute can take NULL values except for the primary key.

© NCERT not to be republished

Let us identify data types of the attributes of table STUDENT along with their constraint, if any. Assuming maximum students in a class to be 100 and values of roll number in a sequence from 1 to 100, we know that 3 digits are sufficient to store values for the attribute RollNumber. Hence, data type INT is appropriate for this attribute. Total number of characters in student names (SName) can differ. Assuming maximum characters in a name as 20, we use VARCHAR(20) for SName column. Data type for the attribute SDateofBirth is DATE and supposing the school uses guardian's 12 digit Aadhaar number as GUID, we can declare GUID as CHAR(12) since Aadhaar number is of fixed length and we are not going to perform any mathematical operation on GUID.

Table 8.3, 8.4 and 8.5 show the chosen data type and constraint for each attribute of the relations STUDENT, GUARDIAN and ATTENDANCE, respectively.

Table 8.3 Data types and constraints for the attributes of relation STUDENT

Attribute Name	Data expected to be stored	Data type	Constraint
RollNumber	Numeric value consisting of maximum 3 digits	INT	PRIMARY KEY
SName	Variant length string of maximum 20 characters	VARCHAR(20)	NOT NULL
SDateofBirth	Date value	DATE	NOT NULL
GUID	Numeric value consisting of 12 digits	CHAR(12)	FOREIGN KEY

Table 8.4 Data types and constraints for the attributes of relation GUARDIAN

Attribute Name	Data expected to be stored	Data type	Constraint
GUID	Numeric value consisting of 12 digit Aadhaar number	CHAR(12)	PRIMARY KEY
GName	Variant length string of maximum 20 characters	VARCHAR(20)	NOT NULL
GPhone	Numeric value consisting of 10 digits	CHAR(10)	NULL UNIQUE
GAddress	Variant length string of size 30 characters	VARCHAR(30)	NOT NULL

Table 8.5 Data types and constraints for the attributes of relation ATTENDANCE.

Attribute Name	Data expected to be stored	Data type	Constraint
AttendanceDate	Date value	DATE	PRIMARY KEY*
RollNumber	Numeric value consisting of maximum 3 digits	INT	PRIMARY KEY* FOREIGN KEY
AttendanceStatus	'P' for present and 'A' for absent	CHAR(1)	NOT NULL

**means part of composite primary key*

Once data types and constraints are identified, let us create tables without specifying constraint along with the attribute name for simplification. We will learn to incorporate constraints on attributes in Section 8.4.4.

© NCERT not to be republished

Example 8.1 Create table STUDENT.

```
mysql> CREATE TABLE STUDENT(
    -> RollNumber  INT,
    -> SName VARCHAR(20),
    -> SDateofBirth DATE,
    -> GUID CHAR(12),
    -> PRIMARY KEY (RollNumber));
Query OK, 0 rows affected (0.91 sec)
```

Note: ',' is used to separate two attributes and each statement terminates with a semi-colon (;). The symbol '->' indicates line continuation as SQL statement may not complete in a single line.

Think and Reflect

Can we have a CHAR or VARCHAR data type for contact number (mobile, landline)?

Activity 8.4

Create the other two relations GUARDIAN and ATTENDANCE as per data types given in Table 8.4 and 8.5, and view their structures. Don't add any constraint in the two tables.

8.4.3 DESCRIBE Table

We can view the structure of an already created table using the describe statement.

Syntax:

```
DESCRIBE tablename;
```

MySQL also supports the short form DESC of DESCRIBE to get description of table. To retrieve details about the structure of relation STUDENT, we can write DESC or DESCRIBE followed by table name:

```
mysql> DESC STUDENT;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| RollNumber   | int         | NO   | PRI | NULL    |       |
| SName        | varchar(20) | YES  |     | NULL    |       |
| SDateofBirth | date        | YES  |     | NULL    |       |
| GUID         | char(12)    | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.06 sec)
```

The show table command will now return the table STUDENT:

```
mysql> SHOW TABLES;
+-----------------------------+
| Tables_in_studentattendance |
+-----------------------------+
| student                     |
+-----------------------------+
1 row in set (0.00 sec)
```

8.4.4 ALTER Table

After creating a table we may realize that we need to add/remove an attribute or to modify the datatype of an existing attribute or to add constraint in attribute. In all such cases, we need to change or alter the structure of the table by using the alter statement.

Syntax:

```
ALTER TABLE tablename ADD/Modify/DROP attribute1,
attribute2,..
```

© NCERT not to be republished

(A) Add primary key to a relation

Let us now alter the tables created in Activity 8.4. The below MySQL statement adds a primary key to the GUARDIAN relation:

```
mysql> ALTER TABLE GUARDIAN ADD PRIMARY KEY (GUID);
Query OK, 0 rows affected (1.14 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

Now let us add primary key to the ATTENDANCE relation. The primary key of this relation is a composite key made up of two attributes — AttendanceDate and RollNumber.

```
mysql> ALTER TABLE ATTENDANCE
    -> ADD PRIMARY KEY(AttendanceDate,
    -> RollNumber);
Query OK, 0 rows affected (0.52 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(B) Add foreign key to a relation

Once primary keys are added the next step is to add foreign keys to the relation (if any). A relation may have multiple foreign keys and each foreign key is defined on a single attribute. Following points need to be observed while adding foreign key to a relation:

- The referenced relation must be already created.
- The referenced attribute must be a part of primary key of the referenced relation.
- Data types and size of referenced and referencing attributes must be same.

Syntax:

```
ALTER TABLE table_name ADD FOREIGN KEY(attribute
name) REFERENCES referenced_table_name
(attribute name);
```

Let us now add foreign key to the table STUDENT. Table 8.3 shows that attribute GUID (the referencing attribute) is a foreign key and it refers to attribute GUID (the referenced attribute) of table GUARDIAN (Table 8.4). Hence, STUDENT is the referencing table and GUARDIAN is the referenced table.

Think and Reflect

Name foreign keys in table ATTENDANCE and STUDENT. Is there any foreign key in table GUARDIAN.

```
mysql> ALTER TABLE STUDENT
    -> ADD FOREIGN KEY(GUID) REFERENCES
    -> GUARDIAN(GUID);
Query OK, 0 rows affected (0.75 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(C) Add constraint UNIQUE to an existing attribute

In GUARDIAN table, attribute GPhone has a constraint `UNIQUE` which means no two values in that column should be same.

Syntax:

© NCERT not to be republished

```
ALTER TABLE table_name ADD UNIQUE (attribute
name);
```

Let us now add the constraint UNIQUE with attribute GPhone of the table GUARDIAN as shown at table 8.4.

```
mysql> ALTER TABLE GUARDIAN
    -> ADD UNIQUE(GPhone);
Query OK, 0 rows affected (0.44 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(D) Add an attribute to an existing table

Sometimes, we may need to add an additional attribute in a table. It can be done using the syntax given below:

```
ALTER TABLE table_name ADD attribute_name DATATYPE;
```

Suppose the principal of the school has decided to award scholarship to some needy students for which income of the guardian must be known. But school has not maintained income attribute with table GUARDIAN so far. Therefore, the database designer now needs to add a new attribute income of data type INT in the table GUARDIAN.

```
mysql> ALTER TABLE GUARDIAN
    -> ADD income INT;
Query OK, 0 rows affected (0.47 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(E) Modify datatype of an attribute

We can modify data types of the existing attributes of a table using the following ALTER statement.

Syntax:

```
ALTER TABLE table_name MODIFY attribute DATATYPE;
```

Suppose we need to change the size of attribute GAddress from VARCHAR(30) to VARCHAR(40) of the GUARDIAN table. The MySQL statement will be:

```
mysql> ALTER TABLE GUARDIAN
    -> MODIFY GAddress VARCHAR(40);
Query OK, 0 rows affected (0.11 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(F) Modify constraint of an attribute

When we create a table, by default each attribute takes NULL value except for the attribute defined as primary key. We can change an attribute's constraint from NULL to NOT NULL using alter statement.

Syntax:

```
ALTER TABLE table_name MODIFY attribute DATATYPE
NOT NULL;
```

Note: We have to specify the data type of the attribute along with constraint NOT NULL while using MODIFY.

Activity 8.5

Add foreign key in the ATTENDANCE table (use fig. 8.1 to identify referencing and referenced tables).

Think and Reflect

What are the minimum and maximum income values that can be entered in the income attribute given the data type is INT?

© NCERT not to be republished

NOTES

To associate NOT NULL constraint with attribute SName of table STUDENT (table 8.3), we write the following MySQL statement:

```
mysql> ALTER TABLE STUDENT
    -> MODIFY SName VARCHAR(20) NOT NULL;
Query OK, 0 rows affected (0.47 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(G) Add default value to an attribute

If we want to specify default value for an attribute, then use the following syntax:

```
ALTER TABLE table_name MODIFY attribute DATATYPE
DEFAULT default_value;
```

To set default value of SDateofBirth of STUDENT to 15th May 2000, we write the following statement:

```
mysql> ALTER TABLE STUDENT
    -> MODIFY SDateofBirth DATE DEFAULT
    -> 2000-05-15;
Query OK, 0 rows affected (0.08 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

Note: We have to specify the data type of the attribute along with DEFAULT while using MODIFY.

(H) Remove an attribute

Using ALTER, we can remove attributes from a table, as shown in the below syntax:

```
ALTER TABLE table_name DROP attribute;
```

To remove the attribute income from the table GUARDIAN (8.4), we can write the following MySQL statement:

```
mysql> ALTER TABLE GUARDIAN DROP income;
Query OK, 0 rows affected (0.42 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(I) Remove primary key from the table

While creating a table, we may have specified incorrect primary key. In such case, we need to drop the existing primary key of the table and add a new primary key.

Syntax:

```
ALTER TABLE table_name DROP PRIMARY KEY;
```

To remove primary key of table GUARDIAN (Table 8.4), we write the following MySQL statement:

```
mysql> ALTER TABLE GUARDIAN DROP PRIMARY KEY;
Query OK, 0 rows affected (0.72 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

Note: We have dropped primary key from GUARDIAN table, but each table should have a primary key to maintain uniqueness. Hence, we have to use ADD command to specify primary key for the GUARDIAN table as shown in earlier examples.

© NCERT not to be republished

NOTES

8.4.5 DROP Statement

Sometimes a table in a database or the database itself needs to be removed. We can use `DROP` statement to remove a database or a table permanently from the system. However, one should be very cautious while using this statement as it cannot be undone.

Syntax to drop a table:

```
DROP TABLE table_name;
```

Syntax to drop a database:

```
DROP DATABASE database_name;
```

Cautions:

1) Using the `Drop` statement to remove a database will ultimately remove all the tables within it.
2) `DROP` statement will remove the tables or database created by you. Hence you may apply `DROP` statement at the end of the chapter.

8.5 SQL FOR DATA MANIPULATION

In the previous section, we created the database `StudentAttendance` having three relations STUDENT, GUARDIAN and ATTENDANCE. When we create a table, only its structure is created but the table has no data. To populate records in the table, `INSERT` statement is used. Similarly, table records can be deleted or updated using SQL data manipulation statements.

Data Manipulation using a database means either retrieval (access) of existing data, insertion of new data, removal of existing data or modification of existing data in the database.

8.5.1 INSERTION of Records

`INSERT INTO` statement is used to insert new records in a table. Its syntax is:

```
INSERT INTO tablename
VALUES(value 1, value 2,....);
```

Here, value 1 corresponds to attribute 1, value 2 corresponds to attribute 2 and so on. Note that we need not to specify attribute names in insert statement if there are exactly same number of values in the `INSERT` statement as the total number of attributes in the table.

Caution: While populating records in a table with foreign key, ensure that records in referenced tables are already populated.

© NCERT not to be republished

Let us insert some records in the StudentAttendance database. We shall insert records in the GUARDIAN table first as it does not have any foreign key. We are going to insert the records given in Table 8.6.

Table 8.6 Records to be inserted into the GUARDIAN table

GUID	GName	GPhone	GAddress
444444444444	Amit Ahuja	5711492685	G-35, Ashok Vihar, Delhi
111111111111	Baichung Bhutia	3612967082	Flat no. 5, Darjeeling Appt., Shimla
101010101010	Himanshu Shah	4726309212	26/77, West Patel Nagar, Ahmedabad
333333333333	Danny Dsouza		S -13, Ashok Village, Daman
466444444666	Sujata P.	3801923168	HNO-13, B- block, Preet Vihar, Madurai

The below statement inserts the first record in the table.

```
mysql> INSERT INTO GUARDIAN
    -> VALUES (444444444444, 'Amit Ahuja',
    -> 5711492685, 'G-35,Ashok vihar, Delhi' );
Query OK, 1 row affected (0.01 sec)
```

We can use the SQL statement SELECT * from table_name to view the inserted records. The SELECT statement will be explained in next section.

```
mysql> SELECT * from  GUARDIAN;
+--------------+-----------------+-----------+------------------------------+
| GUID         | GName           | Gphone    | GAddress                     |
+--------------+-----------------+-----------+------------------------------+
| 444444444444 | Amit Ahuja      | 5711492685 | G-35, Ashok vihar, Delhi     |
+--------------+-----------------+-----------+------------------------------+
1 row in set (0.00 sec)
```

If we want to provide values only for some of the attributes in a table (supposing other attributes having NULL or any other default value), then we shall specify the attribute name alongside each data value as shown in the following syntax of INSERT INTO statement.

Syntax:

```
INSERT INTO tablename (column1, column2, ...)
VALUES (value1, value2,  ...);
```

To insert the fourth record of Table 8.6 where GPhone is not given, we need to insert values in the other three fields (GPhone was set to NULL by default at the time of table creation). In this case, we have to specify the names of attributes in which we want to insert values. The values must be given in the same order in which attributes are written in INSERT command.

```
mysql> INSERT INTO GUARDIAN(GUID, GName, GAddress)
    -> VALUES (333333333333, 'Danny Dsouza',
```

Activity 8.6

Write SQL statements to insert the remaining 3 rows of table 8.6 in table GUARDIAN.

© NCERT not to be republished

```
    -> 'S -13, Ashok Village, Daman' );
Query OK, 1 row affected (0.03 sec)
```

Note: Text and date values must be enclosed in ‘ ’ (single quotes).

```
mysql> SELECT * from GUARDIAN;
+--------------+--------------+------------+----------------------------------+
| GUID         | GName        | Gphone     | GAddress                         |
+--------------+--------------+------------+----------------------------------+
| 333333333333 | Danny Dsouza | NULL       | S -13, Ashok Village, Daman      |
| 444444444444 | Amit Ahuja   | 5711492685 | G-35, Ashok vihar, Delhi         |
+--------------+--------------+------------+----------------------------------+
2 rows in set (0.00 sec)
```

Let us now insert the records given in Table 8.7 into the STUDENT table.

Table 8.7 Records to be inserted into the STUDENT table

RollNumber	SName	SDateofBirth	GUID
1	Atharv Ahuja	2003-05-15	444444444444
2	Daizy Bhutia	2002-02-28	111111111111
3	Taleem Shah	2002-02-28	
4	John Dsouza	2003-08-18	333333333333
5	Ali Shah	2003-07-05	101010101010
6	Manika P.	2002-03-10	466444444666

To insert the first record of Table 8.7, we write the following MySQL statement

```
mysql> INSERT INTO STUDENT
    -> VALUES(1,'Atharv Ahuja','2003-05-15',
    -> 444444444444);
Query OK, 1 row affected (0.11 sec)
```

OR

```
mysql> INSERT INTO STUDENT (RollNumber, SName,
    -> SDateofBirth, GUID)
    -> VALUES (1,'Atharv Ahuja','2003-05-15',
    -> 444444444444);
Query OK, 1 row affected (0.02 sec)
```

```
mysql> SELECT * from STUDENT;
+------------+--------------+--------------+--------------+
| RollNumber | SName        | SDateofBirth | GUID         |
+------------+--------------+--------------+--------------+
|          1 | Atharv Ahuja | 2003-05-15   | 444444444444 |
+------------+--------------+--------------+--------------+
1 row in set (0.00 sec)
```

Let us now insert the third record of Table 8.7 where GUID is NULL. Recall that GUID is foreign key of this table and thus can take NULL value. Hence, we can put NULL value for GUID and insert the record by using the following statement:

Recall that Date is stored in “YYYY-MM-DD” format.

© NCERT not to be republished

```
mysql> INSERT INTO STUDENT
    -> VALUES(3, 'Taleem Shah','2002-02-28',
    -> NULL);
Query OK, 1 row affected (0.05 sec)

mysql> SELECT * from STUDENT;
+------------+-------------+-------------+-------------+
| RollNumber | SName       | SDateofBirth | GUID        |
+------------+-------------+-------------+-------------+
|          1 | Atharv Ahuja | 2003-05-15   | 444444444444 |
|          3 | Taleem Shah  | 2002-02-28   |         NULL |
+------------+-------------+-------------+-------------+
2 rows in set (0.00 sec)
```

We had to write NULL in the above MySQL statement because when not giving the column names, we need to give values for all the columns. Otherwise, we have to give names of attributes along with the values if we need to insert data only for certain attributes, as shown in the next query:

Activity 8.7

Write SQL statements to insert the remaining 4 rows of table 8.7 in table STUDENT.

```
mysql> INSERT INTO STUDENT (RollNumber, SName,
    -> SDateofBirth) VALUES (3, 'Taleem Shah','
    -> 2002-02-28');
Query OK, 1 row affected (0.05 sec)
```

In the above statement we are informing DBMS to insert the corresponding values for the mentioned columns and GUID would be assigned NULL value.

```
mysql> SELECT * from STUDENT;
+------------+-------------+-------------+-------------+
| RollNumber | SName       | SDateofBirth | GUID        |
+------------+-------------+-------------+-------------+
|          1 | Atharv Ahuja | 2003-05-15   | 444444444444 |
|          3 | Taleem Shah  | 2002-02-28   |         NULL |
+------------+-------------+-------------+-------------+
2 rows in set (0.00 sec)
```

Think and Reflect

- Which of the above syntax should be used when we are not sure of the order (with respect to the column) in which the values are to be inserted in the table?
- Can we insert two records with the same roll number?

8.6 SQL FOR DATA QUERY

So far we have learnt how to create database as well as to store and manipulate data. We are interested to store data in a database as it is easier to retrieve data in future from databases in whatever way we want. The Structured Query Language (SQL) has efficient mechanisms to retrieve data stored in multiple tables in a MySQL database (or any other RDBMS). The user enters the SQL commands called queries where the specific requirements for data to be retrieved are provided. The SQL statement SELECT is used to retrieve data from the tables in a database and is also called query statement.

© NCERT not to be republished

8.6.1 SELECT Statement

The SQL statement SELECT is used to retrieve data from the tables in a database and the output is also displayed in tabular form.

Syntax:

```
SELECT attribute1, attribute2, ...
FROM table_name
WHERE condition
```

Here, attribute1, attribute2, ... are the column names of the table table_name from which we want to retrieve data. The FROM clause is always written with SELECT clause as it specifies the name of the table from which data is to be retrieved. The WHERE clause is optional and is used to retrieve data that meet specified condition(s).

Example 8.2 To display the name and date of birth of student with roll number 2, we write the following query:

```
mysql> SELECT SName, SDateofBirth
    -> FROM STUDENT
    -> WHERE RollNumber = 1;
+--------------+--------------+
| SName        | SDateofBirth |
+--------------+--------------+
| Atharv Ahuja | 2003-05-15   |
+--------------+--------------+
1 row in set (0.03 sec)
```

Think and Reflect

Can you think of examples from daily life where storing and querying data in a database can be helpful?

8.6.2 QUERYING using Database OFFICE

Different organisations maintain databases to store data in the form of tables. Let us consider the database OFFICE of an organisation that has many related tables like EMPLOYEE, DEPARTMENT and so on. Every EMPLOYEE in the database is assigned to a DEPARTMENT and his/her Department number (DeptId) is stored as a foreign key in the table EMPLOYEE. Let us consider some data for the table 'EMPLOYEE' as shown in Table 8.8 and apply the SELECT statement to retrieve data:

Table 8.8 EMPLOYEE

EmpNo	Ename	Salary	Bonus	DeptId
101	Aaliya	10000	234	D02
102	Kritika	60000	123	D01
103	Shabbir	45000	566	D01
104	Gurpreet	19000	565	D04
105	Joseph	34000	875	D03

© NCERT not to be republished

NOTES

106	Sanya	48000	695	D02
107	Vergese	15000		D01
108	Nachaobi	29000		D05
109	Daribha	42000		D04
110	Tanya	50000	467	D05

(A) Retrieve selected columns

The following query displays employee numbers of all the employees:

```
mysql> SELECT  EmpNo
    -> FROM  EMPLOYEE;
+-------+
| EmpNo |
+-------+
|   101 |
|   102 |
|   103 |
|   104 |
|   105 |
|   106 |
|   107 |
|   108 |
|   109 |
|   110 |
+-------+
10 rows in set (0.41 sec)
```

To display the employee number and employee name of all the employees, we write the following query:

```
mysql> SELECT EmpNo, Ename
    -> FROM  EMPLOYEE;
+-------+----------+
| EmpNo | Ename    |
+-------+----------+
|   101 | Aaliya   |
|   102 | Kritika  |
|   103 | Shabbir  |
|   104 | Gurpreet |
|   105 | Joseph   |
|   106 | Sanya    |
|   107 | Vergese  |
|   108 | Nachaobi |
|   109 | Daribha  |
|   110 | Tanya    |
+-------+----------+
10 rows in set (0.00 sec)
```

(B) Renaming of columns

In case we want to rename any column while displaying the output, we can do so by using alias `'AS'` in the query as:

Display Employee name as Name in the output for all the employees.

```
mysql> SELECT  EName  AS Name
```

© NCERT not to be republished

NOTES

```
    -> FROM  EMPLOYEE;
+----------+
| Name     |
+----------+
| Aaliya   |
| Kritika  |
| Shabbir  |
| Gurpreet |
| Joseph   |
| Sanya    |
| Vergese  |
| Nachaobi |
| Daribha  |
| Tanya    |
+----------+
10 rows in set (0.00 sec)
```

Example 8.3 Display names of all employees along with their annual salary (Salary*12). While displaying query result, rename EName as Name.

```
mysql> SELECT EName AS Name, Salary*12
    -> FROM EMPLOYEE;
+----------+-----------+
| Name     | Salary*12 |
+----------+-----------+
| Aaliya   |    120000 |
| Kritika  |    720000 |
| Shabbir  |    540000 |
| Gurpreet |    228000 |
| Joseph   |    408000 |
| Sanya    |    576000 |
| Vergese  |    180000 |
| Nachaobi |    348000 |
| Daribha  |    504000 |
| Tanya    |    600000 |
+----------+-----------+
10 rows in set (0.02 sec)
```

Observe that in the output, Salary*12 is displayed as the column name for the annual salary column. In the output table, we can use alias to rename that column as Annual Salary as shown below:

```
mysql> SELECT Ename AS Name, Salary*12 AS
    -> 'Annual Salary'
    -> FROM EMPLOYEE;
+----------+---------------+
| Name     | Annual Salary |
+----------+---------------+
| Aaliya   |        120000 |
| Kritika  |        720000 |
| Shabbir  |        540000 |
| Gurpreet |        228000 |
| Joseph   |        408000 |
| Sanya    |        576000 |
| Vergese  |        180000 |
| Nachaobi |        348000 |
| Daribha  |        504000 |
| Tanya    |        600000 |
+----------+---------------+
10 rows in set (0.00 sec)
```

© NCERT not to be republished

NOTES

Note:

i) Annual Salary will not be added as a new column in the database table. It is just for displaying the output of the query.

ii) If an aliased column name has space as in the case of Annual Salary, it should be enclosed in quotes as 'Annual Salary'.

(C) DISTINCT Clause

By default, SQL shows all the data retrieved through query as output. However, there can be duplicate values. The `SELECT` statement when combined with `DISTINCT` clause, returns records without repetition (distinct records). For example, while retrieving employee's department number, there can be duplicate values as many employees are assigned to same department. To display unique department number for all the employees, we use `DISTINCT` as shown below:

```
mysql> SELECT DISTINCT DeptId
    -> FROM EMPLOYEE;
+--------+
| DeptId |
+--------+
| D02    |
| D01    |
| D04    |
| D03    |
| D05    |
+--------+
5 rows in set (0.03 sec)
```

(D) WHERE Clause

The `WHERE` clause is used to retrieve data that meet some specified conditions. In the OFFICE database, more than one employee can have the same salary. To display distinct salaries of the employees working in the department number D01, we write the following query in which the condition to select the employee whose department number is D01 is specified using the `WHERE` clause:

```
mysql> SELECT DISTINCT Salary
    -> FROM EMPLOYEE
    -> WHERE Deptid='D01';
```

As the column DeptId is of string type, its values are enclosed in quotes ('D01').

```
+--------+
| Salary |
+--------+
|  60000 |
|  45000 |
|  15000 |
+--------+
3 rows in set (0.02 sec)
```

© NCERT not to be republished

In the above example, we have used = operator in WHERE clause. We can also use other relational operators (<, <=, >, >=, !=) to specify conditions. The logical operators AND, OR, and NOT are used with WHERE clause to combine multiple conditions.

Example 8.4 Display all the employees who are earning more than 5000 and work in department with DeptId D04.

```
mysql> SELECT  *
    -> FROM  EMPLOYEE
    -> WHERE Salary > 5000 AND DeptId = 'D04';
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   104 | Gurpreet |  19000 |   565 | D04    |
|   109 | Daribha  |  42000 |  NULL | D04    |
+-------+----------+--------+-------+--------+
2 rows in set (0.00 sec)
```

Example 8.5 The following query displays records of all the employees except Aaliya.

```
mysql> SELECT *
    -> FROM EMPLOYEE
    -> WHERE NOT Ename = 'Aaliya';
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   102 | Kritika  |  60000 |   123 | D01    |
|   103 | Shabbir  |  45000 |   566 | D01    |
|   104 | Gurpreet |  19000 |   565 | D04    |
|   105 | Joseph   |  34000 |   875 | D03    |
|   106 | Sanya    |  48000 |   695 | D02    |
|   107 | Vergese  |  15000 |  NULL | D01    |
|   108 | Nachaobi |  29000 |  NULL | D05    |
|   109 | Daribha  |  42000 |  NULL | D04    |
|   110 | Tanya    |  50000 |   467 | D05    |
+-------+----------+--------+-------+--------+
9 rows in set (0.00 sec)
```

Think and Reflect

What will happen if in the above query we write "Aaliya" as "AALIYA" or "aaliya" or "AaLIYA"? Will the query generate the same output or an error?

Example 8.6 The following query displays name and department number of all those employees who are earning salary between 20000 and 50000 (both values inclusive).

```
mysql> SELECT Ename, DeptId
    -> FROM EMPLOYEE
    -> WHERE Salary>=20000 AND Salary<=50000;
+----------+--------+
| Ename    | DeptId |
+----------+--------+
| Shabbir  | D01    |
| Joseph   | D03    |
| Sanya    | D02    |
| Nachaobi | D05    |
| Daribha  | D04    |
| Tanya    | D05    |
+----------+--------+
6 rows in set (0.00 sec)
```

Activity 8.8

Compare the output produced by the query in example 8.6 and the following query and differentiate between the OR and AND operators.

```
SELECT  *
FROM  EMPLOYEE
WHERE Salary > 5000 OR
      DeptId= 20;
```

© NCERT not to be republished

NOTES

The above query defines a range that can also be checked using a comparison operator BETWEEN.

```
mysql> SELECT  Ename, DeptId
    -> FROM EMPLOYEE
    -> WHERE Salary BETWEEN 20000 AND 50000;
+----------+--------+
| Ename    | DeptId |
+----------+--------+
| Shabbir  | D01    |
| Joseph   | D03    |
| Sanya    | D02    |
| Nachaobi | D05    |
| Daribha  | D04    |
| Tanya    | D05    |
+----------+--------+
6 rows in set (0.03 sec)
```

Note: The BETWEEN operator defines the range of values in which the column value must fall into, to make the condition true.

Example 8.7 The following query displays details of all the employees who are working either in DeptId D01, D02 or D04.

```
mysql> SELECT  *
    -> FROM EMPLOYEE
    -> WHERE DeptId = 'D01' OR DeptId = 'D02' OR
    -> DeptId = 'D04';
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   101 | Aaliya   |  10000 |   234 | D02    |
|   102 | Kritika  |  60000 |   123 | D01    |
|   103 | Shabbir  |  45000 |   566 | D01    |
|   104 | Gurpreet |  19000 |   565 | D04    |
|   106 | Sanya    |  48000 |   695 | D02    |
|   107 | Vergese  |  15000 |  NULL | D01    |
|   109 | Daribha  |  42000 |  NULL | D04    |
+-------+----------+--------+-------+--------+
7 rows in set (0.00 sec)
```

(E) MEMBERSHIP OPERATOR IN

The IN operator compares a value with a set of values and returns true if the value belongs to that set. The above query can be rewritten using IN operator as shown below:

```
mysql> SELECT  *
    -> FROM EMPLOYEE
    -> WHERE DeptId IN ('D01', 'D02' , 'D04');
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   101 | Aaliya   |  10000 |   234 | D02    |
|   102 | Kritika  |  60000 |   123 | D01    |
|   103 | Shabbir  |  45000 |   566 | D01    |
|   104 | Gurpreet |  19000 |   565 | D04    |
|   106 | Sanya    |  48000 |   695 | D02    |
|   107 | Vergese  |  15000 |  NULL | D01    |
|   109 | Daribha  |  42000 |  NULL | D04    |
+-------+----------+--------+-------+--------+
7 rows in set (0.00 sec)
```

© NCERT not to be republished

NOTES

Example 8.8 The following query displays details of all the employees except those working in department number D01 or D02.

```
mysql> SELECT *
    -> FROM EMPLOYEE
    -> WHERE DeptId NOT IN('D01', 'D02');
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   104 | Gurpreet |  19000 |   565 | D04    |
|   105 | Joseph   |  34000 |   875 | D03    |
|   108 | Nachaobi |  29000 |  NULL | D05    |
|   109 | Daribha  |  42000 |  NULL | D04    |
|   110 | Tanya    |  50000 |   467 | D05    |
+-------+----------+--------+-------+--------+
5 rows in set (0.00 sec)
```

Note: Here we need to combine `NOT` with `IN` as we want to retrieve all records except with DeptId D01 and D02.

(F) ORDER BY Clause

`ORDER BY` clause is used to display data in an ordered (arranged) form with respect to a specified column. By default, `ORDER BY` displays records in ascending order of the specified column's values. To display the records in descending order, the `DESC` (means descending) keyword needs to be written with that column.

Example 8.9 The following query displays details of all the employees in ascending order of their salaries.

```
mysql> SELECT  *
    -> FROM EMPLOYEE
    -> ORDER BY Salary;
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   101 | Aaliya   |  10000 |   234 | D02    |
|   107 | Vergese  |  15000 |  NULL | D01    |
|   104 | Gurpreet |  19000 |   565 | D04    |
|   108 | Nachaobi |  29000 |  NULL | D05    |
|   105 | Joseph   |  34000 |   875 | D03    |
|   109 | Daribha  |  42000 |  NULL | D04    |
|   103 | Shabbir  |  45000 |   566 | D01    |
|   106 | Sanya    |  48000 |   695 | D02    |
|   110 | Tanya    |  50000 |   467 | D05    |
|   102 | Kritika  |  60000 |   123 | D01    |
+-------+----------+--------+-------+--------+
10 rows in set (0.05 sec)
```

Example 8.10 The following query displays details of all the employees in descending order of their salaries.

```
mysql> SELECT  *
    -> FROM EMPLOYEE
    -> ORDER BY Salary DESC;
```

© NCERT not to be republished

```
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   102 | Kritika  |  60000 |   123 | D01    |
|   110 | Tanya    |  50000 |   467 | D05    |
|   106 | Sanya    |  48000 |   695 | D02    |
|   103 | Shabbir  |  45000 |   566 | D01    |
|   109 | Daribha  |  42000 |  NULL | D04    |
|   105 | Joseph   |  34000 |   875 | D03    |
|   108 | Nachaobi |  29000 |  NULL | D05    |
|   104 | Gurpreet |  19000 |   565 | D04    |
|   107 | Vergese  |  15000 |  NULL | D01    |
|   101 | Aaliya   |  10000 |   234 | D02    |
+-------+----------+--------+-------+--------+
10 rows in set (0.00 sec)
```

(G) Handling NULL Values

SQL supports a special value called NULL to represent a missing or unknown value. For example, the village column in a table called address will have no value for cities. Hence, NULL is used to represent such unknown values. It is important to note that NULL is different from 0 (zero). Also, any arithmetic operation performed with NULL value gives NULL. For example: 5 + NULL = NULL because NULL is unknown hence the result is also unknown. In order to check for NULL value in a column, we use IS NULL.

Activity 8.9

Execute the following two queries and find out what will happen if we specify two columns in the ORDER BY clause:

```
SELECT  *
FROM EMPLOYEE
ORDER BY Salary,
Bonus;

SELECT  *
FROM EMPLOYEE
ORDER BY Salary,Bonus
desc;
```

Example 8.11 The following query displays details of all those employees who have not been given a bonus. This implies that the bonus column will be blank.

```
mysql> SELECT *
    -> FROM EMPLOYEE
    -> WHERE Bonus IS NULL;
+-------+----------+--------+-------+--------+
| EmpNo | Ename    | Salary | Bonus | DeptId |
+-------+----------+--------+-------+--------+
|   107 | Vergese  |  15000 |  NULL | D01    |
|   108 | Nachaobi |  29000 |  NULL | D05    |
|   109 | Daribha  |  42000 |  NULL | D04    |
+-------+----------+--------+-------+--------+
3 rows in set (0.00 sec)
```

Example 8.12 The following query displays names of all the employees who have been given a bonus. This implies that the bonus column will not be blank.

```
mysql> SELECT EName
    -> FROM EMPLOYEE
    -> WHERE Bonus IS NOT NULL;
+----------+
| EName    |
+----------+
| Aaliya   |
| Kritika  |
| Shabbir  |
| Gurpreet |
| Joseph   |
| Sanya    |
| Tanya    |
+----------+
7 rows in set (0.00 sec)
```

© NCERT not to be republished

NOTES

(H) Substring pattern matching

Many a times we come across situations where we don't want to query by matching exact text or value. Rather, we are interested to find matching of only a few characters or values in column values. For example, to find out names starting with 'T' or to find out pin codes starting with '60'. This is called substring pattern matching. We cannot match such patterns using = operator as we are not looking for exact match. SQL provides `LIKE` operator that can be used with `WHERE` clause to search for a specified pattern in a column.

The `LIKE` operator makes use of the following two wild card characters:

- % (percent)— used to represent zero, one, or multiple characters
- _ (underscore)— used to represent a single character

Example 8.13 The following query displays details of all those employees whose name starts with 'K'.

```
mysql> SELECT  *
    -> FROM EMPLOYEE
    -> WHERE Ename LIKE 'K%';

+-------+---------+--------+-------+--------+
| EmpNo | Ename   | Salary | Bonus | DeptId |
+-------+---------+--------+-------+--------+
|   102 | Kritika |  60000 |   123 | D01    |
+-------+---------+--------+-------+--------+
1 row in set (0.00 sec)
```

Example 8.14 The following query displays details of all those employees whose name ends with 'a'.

```
mysql> SELECT  *
    -> FROM EMPLOYEE
    -> WHERE Ename LIKE '%a';

+-------+---------+--------+-------+--------+
| EmpNo | Ename   | Salary | Bonus | DeptId |
+-------+---------+--------+-------+--------+
|   101 | Aaliya  |  10000 |   234 | D02    |
|   102 | Kritika |  60000 |   123 | D01    |
|   106 | Sanya   |  48000 |   695 | D02    |
|   109 | Daribha |  42000 |  NULL | D04    |
|   110 | Tanya   |  50000 |   467 | D05    |
+-------+---------+--------+-------+--------+
5 rows in set (0.00 sec)
```

Example 8.15 The following query displays details of all those employees whose name consists of exactly 5 letters and starts with any letter but has 'ANYA' after that.

```
mysql> SELECT  *
```

© NCERT not to be republished

```
    -> FROM EMPLOYEE
    -> WHERE Ename LIKE '_ANYA';
+-------+-------+--------+-------+--------+
| EmpNo | Ename | Salary | Bonus | DeptId |
+-------+-------+--------+-------+--------+
|   106 | Sanya |  48000 |   695 | D02    |
|   110 | Tanya |  50000 |   467 | D05    |
+-------+-------+--------+-------+--------+
2 rows in set (0.00 sec)
```

Think and Reflect

When we type first letter of a contact name in our contact list in our mobile phones all the names containing that character are displayed. Can you relate SQL statement with the process? List other real life situations where you can visualize an SQL statement in operation.

Example 8.16 The following query displays names of all the employees containing 'se' as a substring in name.

```
mysql> SELECT  Ename
    -> FROM EMPLOYEE
    -> WHERE Ename LIKE '%se%';
+---------+
| Ename   |
+---------+
| Joseph  |
| Vergese |
+---------+
2 rows in set (0.00 sec)
```

Example 8.17 The following query displays names of all employees containing 'a' as the second character.

```
mysql> SELECT  EName
    -> FROM EMPLOYEE
    -> WHERE Ename LIKE '_a%';
+----------+
| EName    |
+----------+
| Aaliya   |
| Sanya    |
| Nachaobi |
| Daribha  |
| Tanya    |
+----------+
5 rows in set (0.00 sec)
```

8.7 DATA UPDATION AND DELETION

Updation and deletion of data are also the parts of SQL data manipulation. In this section, we are going to apply these two data manipulation methods.

8.7.1 Data Updation

We may need to make changes in the value(s) of one or more columns of existing records in a table. For example, we may require some changes in address, phone number or spelling of name, etc. The `UPDATE` statement is used to make such modifications in the existing data.

Syntax:

```
UPDATE table_name
SET attribute1 = value1, attribute2 = value2, ...
```

© NCERT not to be republished

```
WHERE condition;
```

The STUDENT Table 8.7 has NULL value for GUID for student with roll number 3. Also, suppose students with roll numbers 3 and 5 are siblings. So, in STUDENT table, we need to fill the GUID value for student with roll number 3 as 101010101010. In order to update or change GUID of a particular row (record), we need to specify that record using WHERE clause, as shown below:

```
mysql> UPDATE STUDENT
    -> SET GUID = 101010101010
    -> WHERE RollNumber = 3;
Query OK, 1 row affected (0.06 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

We can then verify the updated data using the statement SELECT * FROM STUDENT.

Caution : If we miss the where clause in the UPDATE statement then the GUID of all the records will be changed to 101010101010.

We can also update values for more than one column using the UPDATE statement. Suppose, the guardian (Table 8.6) with GUID 466444444666 has requested to change the Address to 'WZ - 68, Azad Avenue, Bijnour, MP' and Phone number to '4817362092'.

```
mysql> UPDATE GUARDIAN
    -> SET GAddress = 'WZ - 68, Azad Avenue,
    -> Bijnour, MP', GPhone = 9010810547
    -> WHERE  GUID = 466444444666;
Query OK, 1 row affected (0.06 sec)
Rows matched: 1 Changed: 1 Warnings: 0
mysql> SELECT * FROM GUARDIAN ;
```

```
+------------+--------------+----------+-----------------------------------+
|GUID        |GName         |Gphone    |GAddress                           |
+------------+--------------+----------+-----------------------------------+
|444444444444|Amit Ahuja    |5711492685|G-35, Ashok vihar, Delhi           |
|111111111111|Baichung Bhutia|3612967082|Flat no. 5, Darjeeling Appt., Shimla|
|101010101010|Himanshu Shah |4726309212|26/77, West Patel Nagar, Ahmedabad |
|333333333333|Danny Dsouza  |NULL      |S -13, Ashok Village, Daman        |
|466444444666|Sujata P.     |3801923168|WZ - 68, Azad Avenue, Bijnour, MP  |
+------------+--------------+----------+-----------------------------------+
5 rows in set (0.00 sec)
```

8.7.2 Data Deletion

The DELETE statement is used to delete one or more record(s) from a table.

Syntax:

```
DELETE FROM table_name
WHERE condition;
```

© NCERT not to be republished

Suppose the student with roll number 2 has left the school. We can use the following MySQL statement to delete that record from the STUDENT table.

```
mysql> DELETE FROM STUDENT WHERE RollNumber = 2;
Query OK, 1 row affected (0.06 sec)
```

```
mysql> SELECT * FROM STUDENT ;
+------------+--------------+--------------+--------------+
| RollNumber | SName        | SDateofBirth | GUID         |
+------------+--------------+--------------+--------------+
|          1 | Atharv Ahuja | 2003-05-15   | 444444444444 |
|          3 | Taleem Shah  | 2002-02-28   | 101010101010 |
|          4 | John Dsouza  | 2003-08-18   | 333333333333 |
|          5 | Ali Shah     | 2003-07-05   | 101010101010 |
|          6 | Manika P.    | 2002-03-10   | 466444444666 |
+------------+--------------+--------------+--------------+
5 rows in set (0.00 sec)
```

Caution: Like `UPDATE` statement, we need to be careful to include `WHERE` clause while using `DELETE` statement to delete records in a table. Otherwise, all the records in the table will get deleted.

SUMMARY

- Database is a collection of related tables. MySQL is a 'relational' DBMS. A table is a collection of rows and columns, where each row is a record and columns describe the feature of records.
- SQL is the standard language for most RDBMS. SQL is case insensitive.
- `CREATE DATABASE` statement is used to create a new database.
- `USE` statement is used for making the specified database as active database.
- `CREATE TABLE` statement is used to create a table.
- Every attribute in a `CREATE TABLE` statement must have a name and a datatype.
- `ALTER TABLE` statement is used to make changes in the structure of a table like adding, removing or changing datatype of column(s).
- The `DESC` statement with table name shows the structure of the table.
- `INSERT INTO` statement is used to insert record(s) in a table.
- `UPDATE` statement is used to modify existing data in a table.
- `DELETE` statement is used to delete records in a table.

© NCERT not to be republished

NOTES

- The SELECT statement is used to retrieve data from one or more database tables.
- SELECT * FROM table_name displays data from all the attributes of that table.
- The WHERE clause is used to enforce condition(s) in a query.
- DISTINCT clause is used to eliminate repetition and display the values only once.
- The BETWEEN operator defines the range of values inclusive of boundary values.
- The IN operator selects values that match any value in the given list of values.
- NULL values can be tested using IS NULL and IS NOT NULL.
- ORDER BY clause is used to display the result of an SQL query in ascending or descending order with respect to specified attribute values. The default is ascending order.
- LIKE clause is used for pattern matching. % and _ are two wild card characters. The percent (%) symbol is used to represent zero or more characters. The underscore (_) symbol is used to represent a single character.

© NCERT not to be republished

Exercise

1. Match the following clauses with their respective functions.

ALTER	Insert the values in a table
UPDATE	Restrictions on columns
DELETE	Table definition
INSERT INTO	Change the name of a column
CONSTRAINTS	Update existing information in a table
DESC	Delete an existing row from a table
CREATE	Create a database

2. Choose appropriate answer with respect to the following code snippet.

```
CREATE TABLE student (
            name CHAR(30),
```

NOTES

```
                    student_id INT,
                    gender CHAR(1),
                    PRIMARY KEY (student_id)
);
```

a) What will be the degree of student table?
   i) 30
   ii) 1
   iii) 3
   iv) 4

b) What does 'name' represent in the above code snippet?
   i) a table
   ii) a row
   iii) a column
   iv) a database

c) What is true about the following SQL statement?

```
SelecT * fROM student;
```

   i) Displays contents of table 'student'
   ii) Displays column names and contents of table 'student'
   iii) Results in error as improper case has been used
   iv) Displays only the column names of table 'student'

d) What will be the output of following query?

```
INSERT INTO student
VALUES ("Suhana",109,'F'),
VALUES ("Rivaan",102,'M'),
VALUES ("Atharv",103,'M'),
VALUES ("Rishika",105,'F'),
VALUES ("Garvit",104,'M'),
VALUES ("Shaurya",109,'M');
```

   i) Error
   ii) No Error
   iii) Depends on compiler
   iv) Successful completion of the query

e) In the following query how many rows will be deleted?

```
DELETE student
WHERE student_id=109;
```

   i) 1 row
   ii) All the rows where student ID is equal to 109
   iii) No row will be deleted
   iv) 2 rows

3. Fill in the blanks:

a) _____ declares that an index in one table is related to that in another table.

   i) Primary Key
   ii) Foreign Key
   iii) Composite Key
   iv) Secondary Key

b) The symbol Asterisk (*) in a select query retrieves _____.

   i) All data from the table
   ii) Data of primary key only

© NCERT not to be republished

iii) NULL data
iv) None of the mentioned

4. Consider the following MOVIE database and answer the SQL queries based on it.

MovieID	MovieName	Category	ReleaseDate	ProductionCost	BusinessCost
001	Hindi_Movie	Musical	2018-04-23	124500	130000
002	Tamil_Movie	Action	2016-05-17	112000	118000
003	English_Movie	Horror	2017-08-06	245000	360000
004	Bengali_Movie	Adventure	2017-01-04	72000	100000
005	Telugu_Movie	Action	-	100000	-
006	Punjabi_Movie	Comedy	-	30500	-

a) Retrieve movies information without mentioning their column names.
b) List business done by the movies showing only MovieID, MovieName and BusinessCost.
c) List the different categories of movies.
d) Find the net profit of each movie showing its ID, Name and Net Profit.
(***Hint:*** Net Profit = BusinessCost – ProductionCost)
Make sure that the new column name is labelled as NetProfit. Is this column now a part of the MOVIE relation. If no, then what name is coined for such columns? What can you say about the profit of a movie which has not yet released? Does your query result show profit as zero?
e) List all movies with ProductionCost greater than 80,000 and less than 1,25,000 showing ID, Name and ProductionCost.
f) List all movies which fall in the category of Comedy or Action.
g) List the movies which have not been released yet.

5. Suppose your school management has decided to conduct cricket matches between students of class XI and Class XII. Students of each class are asked to join any one of the four teams — Team Titan, Team Rockers, Team Magnet and Team Hurricane. During summer vacations, various matches will be conducted between these teams. Help your sports teacher to do the following:
a) Create a database "Sports".
b) Create a table "TEAM" with following considerations:
i) It should have a column TeamID for storing an integer value between 1 to 9, which refers to unique identification of a team.
ii) Each TeamID should have its associated name (TeamName), which should be a string of length not less than 10 characters.

© NCERT not to be republished

c) Using table level constraint, make TeamID as primary key.

d) Show the structure of the table TEAM using SQL command.

e) As per the preferences of the students four teams were formed as given below. Insert these four rows in TEAM table:
Row 1: (1, Team Titan)
Row 2: (2, Team Rockers)
Row 3: (3, Team Magnet)
Row 4: (4, Team Hurricane)

f) Show the contents of the table TEAM.

g) Now create another table below. MATCH_DETAILS and insert data as shown in table. Choose appropriate domains and constraints for each attribute.

Table: MATCH_DETAILS

MatchID	MatchDate	FirstTeamID	SecondTeamID	FirstTeamScore	SecondTeamScore
M1	2018-07-17	1	2	90	86
M2	2018-07-18	3	4	45	48
M3	2018-07-19	1	3	78	56
M4	2018-07-19	2	4	56	67
M5	2018-07-20	1	4	32	87
M6	2018-07-21	2	3	67	51

h) Use the foreign key constraint in the MATCH_DETAILS table with reference to TEAM table so that MATCH_DETAILS table records score of teams existing in the TEAM table only.

6. Using the sports database containing two relations (TEAM, MATCH_DETAILS), answer the following relational algebra queries.

a) Retrieve the MatchID of all those matches where both the teams have scored > 70.

b) Retrieve the MatchID of all those matches where FirstTeam has scored < 70 but SecondTeam has scored > 70.

c) Find out the MatchID and date of matches played by Team 1 and won by it.

d) Find out the MatchID of matches played by Team 2 and not won by it.

e) In the TEAM relation, change the name of the relation to T_DATA. Also change the attributes TeamID and TeamName to T_ID and T_NAME respectively.

7. Differentiate between the following commands:
a) ALTER and UPDATE
b) DELETE and DROP

© NCERT not to be republished

8. Create a database called STUDENT_PROJECT having the following tables. Choose appropriate data type and apply necessary constraints.

Table: STUDENT

RollNo	Name	Stream	Section	RegistrationID

* The values in Stream column can be either Science, Commerce, or Humanities.
* The values in Section column can be either I or II.

Table: PROJECT_ASSIGNED

RegistrationID	ProjectID	AssignDate

Table: PROJECT

ProjectID	ProjectName	SubmissionDate	TeamSize	GuideTeacher

a) Populate these tables with appropriate data.
b) Write SQL queries for the following.
c) Find the names of students in Science Stream.
d) What will be the primary keys of the three tables?
e) What are the foreign keys of the three relations?
f) Finds names of all the students studying in class 'Commerce stream' and are guided by same teacher, even if they are assigned different projects.

9. An organization ABC maintains a database `EMP-DEPENDENT` to record the following details about its employees and their dependents.

```
EMPLOYEE(AadhaarNo, Name, Address, Department,
EmpID)
DEPENDENT(EmpID, DependentName, Relationship)
```

Use the `EMP-DEPENDENT` database to answer the following SQL queries:

a) Find the names of employees with their dependent names.
b) Find employee details working in a department, say, 'PRODUCTION'.
c) Find employee names having no dependent
d) Find names of employees working in a department, say, 'SALES' and having exactly two dependents.

10. A shop called Wonderful Garments that sells school uniforms maintain a database `SCHOOL_UNIFORM` as shown below. It consisted of two relations — UNIFORM and PRICE. They made UniformCode as the primary key for UNIFORM relation. Further, they used UniformCode and Size as composite keys for PRICE relation. By analysing the database schema and database state, specify SQL queries to rectify the following anomalies.

© NCERT not to be republished

a) The PRICE relation has an attribute named Price. In order to avoid confusion, write SQL query to change the name of the relation PRICE to COST.

UNIFORM

UCode	UName	UColor
1	Shirt	White
2	Pant	Grey
3	Skirt	Grey
4	Tie	Blue
5	Socks	Blue
6	Belt	Blue

PRICE

UCode	Size	Price
1	M	500
1	L	580
1	XL	620
2	M	810
2	L	890
2	XL	940
3	M	770
3	L	830
3	XL	910
4	S	150
4	L	170
5	S	180
5	L	210
6	M	110
6	L	140
6	XL	160

b) M/S Wonderful Garments also keeps handkerchiefs of red color, medium size of ₹100 each. Insert this record in COST table.

c) When you used the above query to insert data, you were able to enter the values for handkerchief without entering its details in the UNIFORM relation. Make a provision so that the data can be entered in COST table only if it is already there in UNIFROM table.

d) Further, you should be able to assign a new UCode to an item only if it has a valid UName. Write a query to add appropriate constraint to the SCHOOL_UNIFORM database.

e) ALTER table to add the constraint that price of an item is always greater than zero.

© NCERT not to be republished